Frank Stenger · Don Tucker
Gerd Baumann

Navier–Stokes Equations on $\mathbb{R}^3 \times [0, T]$

Springer

Navier–Stokes Equations on $\mathbb{R}^3 \times [0, T]$

Frank Stenger • Don Tucker • Gerd Baumann

Navier–Stokes Equations on $\mathbb{R}^3 \times [0, T]$

Springer

Frank Stenger
School of Computing
University of Utah
Salt Lake City, UT, USA

Don Tucker
Department of Mathematics
University of Utah
Salt Lake City, UT, USA

Gerd Baumann
Department of Mathematics
German University in Cairo
New Cairo City, Egypt

University of Ulm
Ulm, Germany

ISBN 978-3-319-27524-6 ISBN 978-3-319-27526-0 (eBook)
DOI 10.1007/978-3-319-27526-0

Library of Congress Control Number: 2016930314

© Springer International Publishing AG 2016
This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.
The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.
The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, express or implied, with respect to the material contained herein or for any errors or omissions that may have been made.

Printed on acid-free paper

This Springer imprint is published by Springer Nature
The registered company is Springer International Publishing AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Preface

In this monograph we study the properties of solutions of the Navier–Stokes (N–S) partial differential equations (PDE) on $(x, y, z, t) \in \mathbb{R}^3 \times (0, T)$. Initially we convert the PDE to a system of integral equations (IE). We then describe spaces $\mathbf{A}$ of analytic functions that house solutions of this equation, and we show that these spaces of analytic functions are dense in the spaces $\mathbf{S}$ of rapidly decreasing and infinitely differentiable functions. These spaces are defined more explicitly later in this monograph. Some reasons for doing this are the following:

1. The functions of $\mathbf{S}$ are nearly always conceptual rather than explicit, i.e., relatively few such explicit functions are known, and except in concept, they differ from functions of calculus, which are generally analytic.
2. Initial and boundary conditions of solutions of PDE are usually given by scientists of applications, and as such, they are nearly always piecewise analytic, and in this case the solutions have the same properties.
3. When methods of approximation are applied to functions of $\mathbf{A}$, they converge at an exponential rate, whereas methods of approximation applied to the functions of $\mathbf{S}$ converge only at a polynomial rate.
4. The space $\mathbf{A}$ also provides other conveniences, such as enabling sharper bounds on the solution, enabling easier existence proofs, and enabling a more accurate and more efficient method of solution including accurate error bounds—all of which are included in this monograph.

Following our proofs of denseness, we prove the existence of a solution of the IE in the space of functions $\mathbf{A} \cap \mathbb{R}^3 \times (0, T)$, and we provide an explicit novel algorithm based on Sinc approximation and Picard-like iteration for computing the solution.

We also provide an explicit *Mathematica* program for computing the solution based on our approximation procedure, given the initial divergence-free velocity, and we provide explicit illustrations of our computed solution.

More specifically, the problem which we shall analyze and solve numerically in this monograph is the PDE problem as described by Fefferman [1] for the space $\mathbf{S}$, i.e.,

$$\frac{\partial u^j}{\partial t} - \varepsilon\, \Delta\, u^j = -\sum_{k=1}^{3} u^k \frac{\partial u^j}{\partial x^k} - \frac{\partial p}{\partial x^j}, \quad (\mathbf{r}, t) \in \mathbb{R}^3 \times \mathbb{R}_+. \tag{1}$$

This problem is to be solved subject to the divergence-free condition

$$\operatorname{div} \mathbf{u} = \sum_{k=1}^{3} \frac{\partial u^j}{\partial x^j} = 0, \quad (\bar{r}, t) \in \mathbb{R}^3 \times \mathbb{R}_+, \tag{2}$$

and subject to initial conditions

$$\mathbf{u}^0(\bar{r}) = \mathbf{u}(\bar{r}, 0) \quad \mathbf{r} \in \mathbb{R}^3. \tag{3}$$

Here, $\mathbf{u} = \left(u^1, u^2, u^3\right)$ denotes a velocity vector of flow, p denotes the pressure, $\mathbf{u}^0 = \left(u^{0,1}, u^{0,2}, u^{0,3}\right)$ is a given divergence-free velocity field on $\mathbb{R}^3$, ε is a positive coefficient (the viscosity), and Δ denotes the Laplacian, $\Delta = \sum_{i=1}^{3} \frac{\partial^2}{(\partial x^i)^2}$.

This monograph deals mainly with the solution of the above PDE using its corresponding integral equation formulation, due to recent developments enabling much more efficient and much more rapidly convergent solutions, making it possible for us to obtain solutions over infinite domains [2]. We thus first derive the following integral equation (IE), which can be written in the operator form

$$\mathbf{u} = \mathbf{v} + \mathbf{N}\,\mathbf{u}, \tag{4}$$

where the terms on the right-hand side are defined as follows:

$$\begin{aligned} \mathbf{v}(\bar{r}, t) &= \int_{\mathbb{R}^3} \mathscr{G}(\bar{r} - \bar{r}', t)\, \mathbf{u}^0(\bar{r})\, d\bar{r}' \\ \mathbf{N}\,\mathbf{u}(\bar{r}, t) &= \int_0^t \int_{\mathbb{R}^3} \left\{ \left(\left(\nabla' \mathscr{G}\right) \cdot \mathbf{u}(\bar{r}', t') \right) \mathbf{u}(\bar{r}', t') \right. \\ &\quad \left. + \left(\nabla' \mathscr{G}\right)\, p(\bar{r}', t') \right\} d\bar{r}'\, dt'. \end{aligned} \tag{5}$$

In (5) ∇' indicates the gradient taken with respect to $\bar{r}'$, and we have written $\mathscr{G}$ for $\mathscr{G}(\bar{r} - \bar{r}', t - t')$.

According to [1], the initial condition $\mathbf{u}^0$ must belong to a class of functions $\mathbf{S}^3$ which are infinitely differentiable with respect to all variables and which are rapidly decreasing with respect to each spacial variable on the real line. This class is described in more detail in Sect. 1.2 below. We also introduce in Chap. 2 a class of analytic functions $\mathbf{A}$ which is a subclass of $\mathbf{S}$, which we prove to be dense in the class $\mathbf{S}$, and to which the solutions of (4) belong to whenever $\mathbf{u}^0 \in \mathbf{A}$.

After proving the denseness of the class $\mathbf{A}$ in $\mathbf{S}$, we prove in Chap. 3 that the integral part of the above IE maps functions of $\mathbf{A}$ back into $\mathbf{A}$; in Chap. 4 we prove

that if the initial condition vector $\mathbf{u}^0$ belongs to $\mathbf{A}$ and is divergence-free, then the above (IE) has a solution in $\mathbf{A}$ for all T sufficiently small; in Chap. 5 we introduce an iterative method of solution of the IE; and in Chap. 6 we provide two explicit examples and its numerical solution. Appendix A provides a detailed step-by-step description of our method of solution including an explicit *Mathematica* program based on our explicit algorithmic procedure. Appendix B contains for demonstration purposes an explicit example of a result data file generated by our algorithm.

Salt Lake City, UT, USA — Frank Stenger
Salt Lake City, UT, USA — Don Tucker
Cairo, Egypt — Gerd Baumann
September 19, 2016

References

1. C. Fefferman, Existence and smoothness of the Navier–Stokes equation. Clay Mathematics Institute (2000), http://www.claymath.org\/millennium\/
2. F. Stenger, *Handbook of Sinc Numerical Methods*. A 470–page Tutorial & Matlab Package (CRC Press, Boca Raton, 2011)

Contents

Preface v

1 Introduction, PDE, and IE Formulations 1
- 1.1 Introduction: The PDE Problem 1
- 1.2 The Classes $\mathbf{S}^n$ and $\mathbf{S}^n_T$ 2
- 1.3 The Corresponding IE Problem 2
- 1.4 The Pressure p 4
- 1.5 Modifying the IE 5
- 1.6 The Theoretical Iteration Scheme 6
  - 1.6.1 Von Neumann Iteration 6
  - 1.6.2 Gauss–Seidel Iteration 7

References 8

2 Spaces of Analytic Functions 9
- 2.1 The Spaces $\mathbf{A}^n_{\alpha,d}$ and $\mathbf{A}^n_{\alpha,d,T}$ 9
- 2.2 Denseness of $\mathbf{A}^n_{\alpha,d}$ in $\mathbf{S}^n$ 11
  - 2.2.1 Denseness of Sinc Approximation in $\mathbf{S}^n$ 12
  - 2.2.2 Denseness of the Space $\mathbf{A}^n_{\alpha,d}$ in $\mathbf{S}^n$ 18

References 18

3 Spaces of Solution of the N–S Equations 19
- 3.1 Banach Spaces 19
- 3.2 The Vector $\mathbf{v} = \mathbf{G}\,\mathbf{u}^0$ 20
- 3.3 The Operation $\mathbf{G}_{x^\ell}\mathbf{w}$ 21
- 3.4 The Operation $p = \mathbf{G}_0\, g$ 22
- 3.5 The Operation $\mathbf{K}_{x^\ell}\,\mathbf{w}$ 28
- 3.6 Bilinear Forms for $\mathbf{Nu}$ 28
  - 3.6.1 The Operation $\mathbf{p}(\mathbf{u}, \mathbf{w})$ 29
  - 3.6.2 The Operation $\mathbf{q}(\mathbf{u}, \mathbf{w})$ 30

4 Proof of Convergence of Iteration (1.25) 33
4.1 Contraction Mapping Principle 33

5 Numerical Methods for Solving N–S Equations 37
5.1 Approximating Bases 37
5.1.1 Bases for Solving N–S Equations 37
5.2 Approximation Operations on Bases 38
5.2.1 DFT Approximation 38
5.2.2 Indefinite Integration 40
5.2.3 Differentiation 41
5.2.4 Indefinite Convolution 41
5.3 Approximating the N–S Equations 45
5.3.1 The Vector $\mathbf{v}$ 46
5.3.2 The Approximation of $\mathbf{P}\,\mathbf{u} + \mathbf{Q}\,\mathbf{u}$ 47
5.4 A More Detailed Algorithm 48
References 51

6 Sinc Convolution Examples 53
6.1 Introduction 53
6.2 Example 1: Gaussian Multiplier 56
6.3 Example 2: Hyperbolic Multiplier 72
6.4 Summary 77
References 77

A Implementation Notes 79
A.1 Function Definitions 79
A.1.1 Selection of Initial Values 80
A.1.2 Pressure Potential g 82
A.1.3 Time Steps 84
A.1.4 Spatial Discretization 85
A.1.5 Step Lengths Distribution 88
A.1.6 Discrete Fourier Transform 88
A.1.7 Spatial Temporal Convolution 92
A.1.8 Sinc Functions 96
A.1.9 Conformal Maps 99
A.1.10 Basis Functions 103
A.1.11 Laplace Transform 107
A.1.12 Free Space Green's Function for Poisson's Equation 108
A.1.13 Convolution Integrals 110
A.1.14 Splitted Convolution Integrals 120
A.1.15 Navier–Stokes Function 129
A.1.16 Import Function 144

B Result Notes 147
B.1 Example File of Results 147
References 219

Index 221

Chapter 1
Introduction, PDE, and IE Formulations

Abstract In this chapter we first state the Navier–Stokes (N–S) problem as a system of nonlinear partial differential equations (PDE) along with initial conditions. We then convert this system of PDE to a system of integral equations (IE).

1.1 Introduction: The PDE Problem

The partial differential equation (PDE) problem which we shall consider is the following, as presented in [1], i.e., let $\mathbb{R} = (-\infty, \infty)$ denote the real line, let T denote a positive number, and let the velocity vector $\mathbf{u} = \mathbf{u}(\bar{r}, t) = \left(u^1, u^2, u^3\right)$ satisfy the *Navier–Stokes* (N–S) PDE problem

$$\frac{\partial u^j}{\partial t} - \varepsilon \, \Delta \, u^j = -\sum_{k=1}^{3} u^k \, \frac{\partial u^j}{\partial x_k} - \frac{\partial p}{\partial x_j}, \quad (\bar{r}, t) \in \mathbb{R}^3 \times [0, T]. \tag{1.1}$$

This problem is to be solved subject to the divergence-free condition[1]

$$\operatorname{div} \mathbf{u} = \sum_{j=1}^{3} \frac{\partial u^j}{\partial x_j} = 0, \quad (\bar{r}, t) \in \mathbb{R}^3 \times [0, T], \tag{1.2}$$

and subject to initial conditions

$$\mathbf{u}^0(\bar{r}) = \mathbf{u}(\bar{r}, 0) \quad \bar{r} \in \mathbb{R}^3. \tag{1.3}$$

Here, p denotes the pressure, $\mathbf{u}^0 = \left(u^{0,1}, u^{0,2}, u^{0,3}\right)$ is a given divergence-free velocity field on $\mathbb{R}^3$, ε is a positive coefficient (the viscosity), Δ denotes the Laplacian, $\Delta = \sum_{j=1}^{3} \frac{\partial^2}{(\partial x^j)^2}$, and p denotes the pressure.

[1] Fefferman actually had $[0, \infty)$ instead of $[0, T]$. We shall however consider only the case of $[0, T]$ in this monograph.

© Springer International Publishing AG 2016

F. Stenger et al., *Navier–Stokes Equations on* $\mathbb{R}^3 \times [0, T]$,
DOI 10.1007/978-3-319-27526-0_1

1.2 The Classes $\mathbf{S}^n$ and $\mathbf{S}^n_T$

Let n denote a positive integer, and let $\mathbf{S}^n$ denote the family of all infinitely differentiable functions $g = g(\bar{r})$ defined on $\mathbb{R}^n$ such that

$$|\partial^\alpha_{\bar{r}} g(\bar{r})| \leq C_{\alpha,K}\,(1 + |\bar{r}|)^{-K} \quad \bar{r} \in \mathbb{R}^n, \quad \text{for any constants } \alpha \text{ and } K, \tag{1.4}$$

for some constant $C_{\alpha,K}$ is a constant and similarly, let $\mathbf{S}^n_T$ denote the family of all functions $g = g(\bar{r}, t)$ defined on $\mathbb{R}^n \times [0, T]$, such that for any constants α, K and any integer $m \geq 0$,

$$|\partial^\alpha_{\bar{r}}\, \partial^m_t g(\bar{r})| \leq C_{\alpha;m,K}\,(1 + |\bar{r}|)^{-K}, \quad \bar{r} \in \mathbb{R}^n, \tag{1.5}$$

for some constant $C_{\alpha,m,K}$.

1.3 The Corresponding IE Problem

Let us denote by $\mathscr{G} = \mathscr{G}(\bar{r}, t)$, the well-known Green's function of the heat equation on $\mathbb{R}^3 \times \mathbb{R}_+$, where $\mathbb{R}_+ = [0, \infty)$,

$$\mathscr{G}(\bar{r}, t) = \frac{1}{(4\,\pi\,\varepsilon\,t)^{3/2}}\,\exp\left(-\frac{r^2}{4\,\varepsilon\,t}\right); \quad r = |\bar{r}|. \tag{1.6}$$

This Green's function is the bounded solution to the following initial value problem on $\mathbb{R}^3 \times \mathbb{R}_+$:

$$\begin{gathered} \mathscr{G}_t(\bar{r}, t) - \varepsilon\,\Delta\,\mathscr{G}(\bar{r}, t) = \delta(t)\,\delta^{(3)}(\bar{r}), \\ \mathscr{G}(\bar{r}, 0^+) = 0, \end{gathered} \tag{1.7}$$

where $\delta(t)$ and $\delta^{(3)}$ denote the one- and three-dimensional delta functions, where $\mathbb{R} = (-\infty, \infty)$ and where $\mathbb{R}^+ = (0, \infty)$.

It is straightforward to use this Green's function $\mathscr{G}$ to transform the system (1.1) of three PDE into the following system of three integral equations (IE), for $j = 1, 2, 3$,

$$\begin{aligned} u^j(\bar{r}, t) &= \int_{\mathbb{R}^3} \mathscr{G}(\bar{r} - \bar{r}', t)\,u^{0,j}(\bar{r})\,d\bar{r}' \\ &\quad - \int_0^t \int_{\mathbb{R}^3} \mathscr{G}(\bar{r} - \bar{r}', t - t') \left(\sum_{k=1}^3 u^k\,\frac{\partial u^j}{\partial x'_k} + \frac{\partial p}{\partial x'_j}\right) d\bar{r}'\,dt'. \end{aligned} \tag{1.8}$$

We omit the trivial proof of the following lemma.

Lemma 1.3.1. *If $g \in \mathbf{S}_T^3$, and if γ is defined by either of the integrals*

$$\begin{aligned} \gamma(\bar{r},t) &= \int_{\mathbb{R}^3} \mathscr{G}(\bar{r}-\bar{r}',t)\, g(\bar{r}',t)\, d\bar{r}', \\ \gamma(\bar{r},t) &= \int_0^t \int_{\mathbb{R}^3} \mathscr{G}(\bar{r}-\bar{r}',t-t')\, g(\bar{r}',t')\, d\bar{r}'\, dt', \end{aligned} \tag{1.9}$$

in which $t \in (0,T)$, then $\gamma \in \mathbf{S}_T^3$. Moreover, we have for $j = 1,\ 2,\ 3$, that

$$\begin{aligned} \gamma_{x_j}(\bar{r},t) &= \int_{\mathbb{R}^3} \mathscr{G}_{x_j}(\bar{r}-\bar{r}',t)\, g(\bar{r}',t)\, d\bar{r}' \\ &= -\int_{\mathbb{R}^3} \mathscr{G}_{x_j'}(\bar{r}-\bar{r}',t)\, g(\bar{r}',t)\, d\bar{r}' \\ &= \int_{\mathbb{R}^3} \mathscr{G}(\bar{r}-\bar{r}',t)\, g_{x_j'}(\bar{r}',t)\, d\bar{r}'. \end{aligned} \tag{1.10}$$

Furthermore, the functions γ_{x_j} of (1.10) also belong to $\mathbf{S}_T^3$.

Lemma 1.3.1 enables differing IE expressions for the same system (1.8).

Next, in Theorem 1.3.1 which follows, we present an IE system which is equivalent to the one in (1.8), and with which we shall work within the remainder of this monograph.

Theorem 1.3.1. *If each component of $\mathbf{u}$ belongs to $\mathbf{S}_T^3$, and if $\mathbf{u}$ is divergence-free, then the system of differential equations (1.3) is equivalent to the integral equation formulation*

$$\mathbf{u} = \mathbf{v} + \mathbf{N}\,\mathbf{u}, \tag{1.11}$$

where the terms on the right-hand side are defined as follows:

$$\begin{aligned} \mathbf{v}(\bar{r},t) &= \int_{\mathbb{R}^3} \mathscr{G}(\bar{r}-\bar{r}',t)\, \mathbf{u}^0(\bar{r})\, d\bar{r}' \\ \mathbf{N}\,\mathbf{u}(\bar{r},t) &= \int_0^t \int_{\mathbb{R}^3} \left\{ \left(\left(\nabla' \mathscr{G}\right) \cdot \mathbf{u}(\bar{r}',t') \right) \mathbf{u}(\bar{r}',t') + \right. \\ &\qquad \left. \left(\nabla' \mathscr{G}\right)\ p(\bar{r}',t') \right\} d\bar{r}'\, dt'. \end{aligned} \tag{1.12}$$

In (1.12) ∇' indicates the gradient taken with respect to $\bar{r}'$, and in (1.12), $\mathscr{G} = \mathscr{G}(\bar{r} - \bar{r}', t - t')$.

Proof. Using (1.6), (1.7) as well as Lemma 1.3.1, and assuming that $\mathbf{u}$ is divergence-free, i.e., that $\sum_{j=1}^{3} u^j_{x_j} = 0$, we get the vector IE (1.11).

■

1.4 The Pressure *p*

We here derive an integral expression for the pressure p. Our derived expression is obtained under the assumption that if $\mathbf{u}$ is divergence-free, then so is the vector on the right-hand side of the IE (1.11).

Theorem 1.4.1. *Let each component of $\mathbf{u}^0 \in \mathbf{S}^3$ and let each vector $\mathbf{u} \in \mathbf{S}^3_T$ on the right-hand side of (1.12) be divergence-free. Let the pressure p be selected such that the vector on the right-hand side of (1.12) is also divergence-free. Then p is given by*

$$p(\bar{r}, t) = \int_{\mathbb{R}^3} \mathscr{G}_0(\bar{r} - \bar{r}')\, g(\bar{r}', t)\, d\bar{r}' \tag{1.13}$$

where

$$\begin{aligned} \mathscr{G}_0(\bar{r}) &= \frac{1}{4\pi r}, \\ g(\bar{r}, t) &= \sum_{j=1}^{3} \sum_{k=1}^{3} u^j_{x^k}(\bar{r}, t)\, u^k_{x^j}(\bar{r}, t). \end{aligned} \tag{1.14}$$

Proof. By differentiating each term on the right-hand side of (1.8) with respect to x_j, summing over j and in that way, "forcing" (1.2), we get

$$\sum_{j=1}^{3} u^j_{x_j} = 0 = \sum_{j=1}^{3} \left(A^j - B^j - C^j \right), \tag{1.15}$$

where

$$\begin{aligned} A^j &= \int_{\mathbb{R}^3} \mathscr{G}_{x_j}(\bar{r} - \bar{r}', t)\, u^{0,j}(\bar{r}')\, d\bar{r}' \\ B^j &= \int_0^t \int_{\mathbb{R}^3} \mathscr{G}_{x_j}(\bar{r} - \bar{r}', t - t') \sum_{k=1}^{3} u^k(\bar{r}', t')\, u^j_{x_k}(\bar{r}', t')\, d\bar{r}'\, dt' \\ C^j &= \int_0^t \int_{\mathbb{R}^3} \mathscr{G}_{x_j}(\bar{r} - \bar{r}', t - t')\, p_{x'_j} d\bar{r}'\, dt'. \end{aligned} \tag{1.16}$$

For the term A^j in (1.16) we clearly have $\sum_{j=1}^{3} A^j = 0$, since we assumed that $\mathbf{u}^0$ is divergence-free. Next, by our assumption that the right-hand side of (1.12) is also divergence-free, we must also have $\sum_{j=1}^{3}(B^j + C^j) = 0$. But by application of Lemma 1.3.1 to this sum, we arrive at the equation

$$0 = \int_0^t \int_{\mathbb{R}^3} \mathscr{G}(\bar{r} - \bar{r}', t - t') \left\{ g(\bar{r}', t') + \Delta_{\bar{r}'} p(\bar{r}', t') \right\} d\bar{r}' \, dt', \tag{1.17}$$

in which g is given in (1.14) above.

In this case, because the right-hand side of (1.12) will also be divergence-free we determine $p(\bar{r}, t)$ such that

$$\Delta_{\bar{r}} p(\bar{r}, t) = -g(\bar{r}, t), \quad (\bar{r}, t) \in \mathbb{R}^3 \times [0, T], \tag{1.18}$$

with g given in (1.14). There is only one solution to this PDE problem which satisfies (1.18) and which is bounded on $\mathbb{R}^3$ for all $t \in [0, T]$; this solution is given by the statement of Theorem 1.4.1 above.

■

1.5 Modifying the IE

The above derived expression (1.13) for the pressure p leads to a more complicated kernel in the N–S IE, which we will now simplify.

If we substitute p as given by Theorem 1.4.1 into the IE (1.12), then the resulting right-hand side of the IE involves an integral over $\mathbb{R}^6 \times (0, T)$. We can reduce this integral into a lower dimensional integral based on the result of the following lemma.

Lemma 1.5.1. *Let $\mathscr{K}$ be defined by*

$$\mathscr{K}(\bar{r}, t) = \int_{\mathbb{R}^3} \mathscr{G}(\bar{r} - \bar{r}', t) \, \mathscr{G}_0(\bar{r}') \, d\bar{r}', \tag{1.19}$$

where $\mathscr{G}$ and $\mathscr{G}_0$ are given in (1.6) and (1.14) respectively. Then $\mathscr{K}$ is also given by

$$\mathscr{K}(\bar{r}, t) = \frac{1}{4\pi^{3/2}(\varepsilon t)^{1/2}} \int_0^1 \exp\left(-\frac{r^2}{4\varepsilon t} y^2 \right) dy. \tag{1.20}$$

Proof. We set $\bar{\Lambda} = (\lambda_1, \lambda_2, \lambda_3)$, and $\Lambda = |\bar{\Lambda}|$. It is well known that the Fourier transforms of $\mathscr{G}(\bar{r}, t)$ and $\mathscr{G}_0(\bar{r})$ taken with respect to $\bar{r}$ are $\exp(-\Lambda^2 \varepsilon t)$ and Λ^{-2}

respectively. Hence, it follows from (1.20) that $\mathscr{K}$ is just the inverse transform of the product of these transforms, i.e.,

$$\mathscr{K}(\bar{r},t) = \frac{1}{8\,\pi^3}\int_{\mathbb{R}^3} e^{-i\bar{\Lambda}\cdot\bar{r}}\,\Lambda^{-2}\,\exp(-\Lambda^2\,\varepsilon\,t)\,d\bar{\Lambda}. \tag{1.21}$$

Transforming this integral from Cartesian to spherical coordinates, so that the element of volume $d\bar{\Lambda}$ becomes $\Lambda^2\,d\Omega(\widehat{\bar{\Lambda}})$ where $\widehat{(\cdot)}$ denotes a unit vector and where $d\Omega(\widehat{\bar{\Lambda}})$ denotes the solid angle, and using the easily verifiable result [3],

$$\int_{|\hat{\bar{\Lambda}}|=1} e^{-i\bar{\Lambda}\cdot\bar{r}}\,d\Omega(\bar{\Lambda}) = 4\,\pi\,\frac{\sin(\Lambda\,r)}{\Lambda\,r}, \tag{1.22}$$

that is obtainable by expanding $(\Lambda\,r)^{-1}\,\sin(\Lambda\,r)$ in powers of $\Lambda\,r$ and then performing term wise integration, we arrive at (1.20).

■

1.6 The Theoretical Iteration Scheme

In this section we introduce two iterative methods for solving (1.11)–(1.12), the first, a Neumann-type iterative method, which we shall use to prove existence and uniqueness of the solution, and the second, a Gauss–Seidel type method, which we shall use in Chap. 5 below to get a numerical solution.

1.6.1 Von Neumann Iteration

Let us write the IE (1.11)–(1.12) in the IE form

$$\mathbf{u} = \mathbf{v} + \mathbf{N}\,\mathbf{u}; \quad \mathbf{N}\,\mathbf{u} = \mathbf{P}\,\mathbf{u} + \mathbf{Q}\,\mathbf{u}, \tag{1.23}$$

where $\mathbf{v}$ is given in (1.12), and where

$$\begin{aligned}
&\mathbf{P}\,\mathbf{u}(\bar{r},t)\\
&\quad = \int_0^t \int_{\mathbb{R}^3} \left(\left(\nabla'\,\mathscr{G}(\bar{r}-\bar{r}',t-t')\right)\cdot\mathbf{u}(\bar{r}',t')\right)\mathbf{u}(\bar{r}',t')\,d\bar{r}'\,dt'\\
&\mathbf{Q}\,\mathbf{u}(\bar{r},t) = \int_0^t \int_{\mathbb{R}^3} \left(\left(\nabla'\,\mathscr{K}(\bar{r}-\bar{r}',t-t')\right)\,g(\bar{r}',t')\right)\,d\bar{r}'\,dt'.
\end{aligned} \tag{1.24}$$

In (1.24), g, $\mathscr{G}$, and $\mathscr{K}$ are given in (1.14), (1.6), and (1.19)–(1.20), respectively.

One way to solve (1.23) for sufficiently small T is by use of the classical successive approximation scheme

$$U_{n+1} = \mathbf{v} + \mathbf{N}(U_n), \quad n = 0,\ 1,\ \ldots, \tag{1.25}$$

with $U_0 = \mathbf{v}$. In Chap. 4 we shall prove that this procedure converges on $\mathbb{R}^3 \times [0, T]$ for suitably restricted ranges of T and $\mathbf{v}$.

1.6.2 Gauss–Seidel Iteration

Instead of using the scheme (1.25) to solve (1.12) we shall use the following *Gauss–Seidel*–like scheme,

$$\begin{aligned} u^1_{n+1} &= v^1 + \mathbf{N}^1(u^1_n, u^2_n, u^3_n), \\ u^2_{n+1} &= v^2 + \mathbf{N}^2(u^1_{n+1}, u^2_n, u^3_n), \\ u^3_{n+1} &= v^3 + \mathbf{N}^3(u^1_{n+1}, u^2_{n+1}, u^3_n), \end{aligned} \tag{1.26}$$

starting with $\mathbf{u}_0 = \mathbf{v}$, where

$$\begin{aligned} v^j(\bar{r}, t) &= \int_{\mathbb{R}^3} \mathscr{G}(\bar{r} - \bar{r}', t)\, u^{0,j}(\bar{r}')\, d\bar{r}' \\ \mathbf{N}^j\mathbf{u} &= P^j\mathbf{u} + Q^j\mathbf{u}, \end{aligned} \tag{1.27}$$

and where

$$\begin{aligned} &\mathbf{P}^j\mathbf{u}(\bar{r}, t) \\ &= \int_0^t \int_{\mathbb{R}^3} \left(\nabla' \mathscr{G}(\bar{r} - \bar{r}', t - t') \cdot \mathbf{u}(\bar{r}', t')\right) u^j(\bar{r}', t')\, d\bar{r}'\, dt', \\ &\mathbf{Q}^j(\mathbf{u})(\bar{r}, t) = \int_0^t \int_{\mathbb{R}^3} \mathscr{K}_{x_j}(\bar{r} - \bar{r}', t - t')\, g(\bar{r}', t')\, d\bar{r}'\, dt', \end{aligned} \tag{1.28}$$

where g is given in (1.14). In general, the scheme (1.26) converges more rapidly than the scheme (1.25), and indeed, our programming examples comparing this Gauss–Seidel scheme with the one in [2] bears this out. It thus enables a convergent iterative solution for larger values of T than that for (1.25).

References

1. C. Fefferman, Existence and smoothness of the Navier–Stokes equation. Clay Mathematics Institute (2000), http://www.claymath.org/millennium/
2. F. Stenger, *Handbook of Sinc Numerical Methods*. A 470-page Tutorial & Matlab Package (CRC Press, Boca Raton, 2011)
3. S. Sýkora, K-space images of *n*-dimensional spheres and generalized Sinc functions (2007), www.ebyte.it/library/docs/math07/SincN.html

Chapter 2
Spaces of Analytic Functions

Abstract We present here spaces of analytic functions $\mathbf{A}^n_{\alpha,d} \subset \mathbf{S}^n$ as well as spaces, $\mathbf{A}^n_{\alpha,d,T} \subset \mathbf{S}^n_T$, $n = 1, 2, 3$. In this chapter, we shall study the properties of these spaces, we shall prove in Chap. 3 that if the components of the initial condition vector $\mathbf{u}^0$ belong to $\mathbf{A}^3_{\alpha,d}$ then each component of $\mathbf{N}\,\mathbf{u}$ of (1.23) belongs to $\mathbf{A}^3_{\alpha,d,T}$, and we shall furthermore prove in Chap. 4 that the solution to (1.23) belongs to $\mathbf{A}^3_{\alpha,d,T}$, for all T sufficiently small. These spaces are in fact special cases of the spaces $\mathbf{S}^n$ and $\mathbf{S}^n_T$ introduced in Sect. 1.2. They provide several conveniences, such as enabling sharper error bounds and yielding exponential convergence of our approximate solution which we obtain in Chap. 5.

2.1 The Spaces $\mathbf{A}^n_{\alpha,d}$ and $\mathbf{A}^n_{\alpha,d,T}$

Let n denote a positive integer, and let us now define vector spaces of functions $\mathbf{A}^n_{\alpha,d}$ and $\mathbf{A}^n_{\alpha,d,T}$.

Definition 2.1.1. Set $\bar{r}^* = \bar{r} + i\,\bar{\rho}$, with $\bar{r} \in \mathbb{R}^n$, $\bar{\rho} \in \mathbb{R}^n$, set $r = |\bar{r}|$, and $\rho = |\bar{\rho}|$.

(a) Corresponding to some positive numbers α and d, let $\mathbf{A}_{\alpha,d}$ denote the family of all functions f with the following properties:

(i) *Analyticity property.* There exists a positive number $d' > d$, such that each f is analytic in the domain

$$\mathscr{D}^n_{d'} = \{\bar{r}^* = \bar{r} + i\,\bar{\rho} \in \mathbb{C}^n : \ \rho < d'\}; \tag{2.1}$$

and

(ii) *Asymptotic property.* There exist positive numbers $C = C(f, d')$ and $\alpha' > \alpha$, such that for all $\bar{r}^* \in \mathscr{D}^n_{d'}$,

$$|f(\bar{r}^*)| < C\,\exp(-\alpha'\,r). \tag{2.2}$$

Notice that if f and g belong either to $\mathbf{A}_{\alpha,d}$ or to $\mathbf{A}_{\alpha,d,T}$, then so does h, where for any constants a, b, and c, $h = a f + b\,g$, or $h = c f$.

© Springer International Publishing AG 2016

F. Stenger et al., *Navier–Stokes Equations on* $\mathbb{R}^3 \times [0, T]$,
DOI 10.1007/978-3-319-27526-0_2

(b) We also define the space $\mathbf{A}^n_{\alpha,d,T}$ of functions $f = f(\bar{r}^*, t^*)$ such that $f(\cdot, t) \in \mathbf{A}^n_{\alpha,d}$ for each fixed $t \in [0, T]$, and such that $f(\bar{r}, \cdot)$ is an analytic and uniformly bounded function of $t = t^*$ in the "eye-shaped" region

$$\mathscr{D}_{d',T} = \{t^* \in \mathbb{C} : |\arg(t^*/(T - t^*))| < d'\}. \tag{2.3}$$

(c) The spaces $\mathbf{A}^n_{\alpha,d}$ and $\mathbf{A}^n_{\alpha,d,T}$, are normed for $p \in [1, \infty)$ by $\|\cdot\|_p$, i.e.,

$$\begin{aligned} \|f\|_p &= \left(\int_{\mathbb{R}^n} |f(\bar{r})|^p \, d\bar{r} \right)^{1/p}, \text{ if } p \in [1, \infty) \text{ and} \\ \|f\|_\infty &= \sup_{\bar{r} \in \mathbb{R}^n} |f(\bar{r})|. \end{aligned} \tag{2.4}$$

and similarly,

$$\begin{aligned} \|f\|_{p,T} &= \left(\int_{\mathbb{R}^n} \int_0^T |f(\bar{r}, t)|^p \, d\bar{r} \, dt \right)^{1/p}, \text{ if } p \in [1, \infty) \text{ and} \\ \|f\|_{\infty,T} &= \sup_{\bar{r} \in \mathbb{R}^n, t \in (0,T)} |f(\bar{r}, t)|. \end{aligned} \tag{2.5}$$

The following theorem describes an important and beautiful property of the class of functions $\mathbf{A}^n_{\alpha,d}$.

Theorem 2.1.1. *In the notation of Definition 2.1.1, let $f \in \mathbf{A}^n_{\alpha,d}$. Let $\hat{f}$ denote the Fourier transform of $\hat{f}$, i.e.,*

$$\hat{f}(\bar{\Lambda}) = \int_{\mathbb{R}^n} f(\bar{r}') \, \exp(i \, \bar{\Lambda} \cdot \bar{r}') \, d\bar{r}'. \tag{2.6}$$

Then $\hat{f} \in \mathbf{A}^n_{d,\alpha}$.

Proof. The one-dimensional version of this result is found in Theorem 26 of Sect. 1.27 of [3]. The proof of the three-dimensional case is similar, and we omit it.

■

Upon recalling the inverse Fourier transform formula for $\mathbb{R}^3$ [see, e.g., (1.21)] we also have by Parseval's theorem that

$$\|\hat{f}\|_2 = (2\pi)^{3/2} \, \|f\|_2, \tag{2.7}$$

and similarly for $\|\hat{f}\|_{2,T}$.

Definition 2.1.2. Let n denote a positive integer, and set $\mathbf{b} = (b_1, \ldots, b_n)$, with b_j nonnegative integers and with $|\mathbf{b}| = \sum_{j=1}^n b_j$, and define

$$D^{\mathbf{b}} f = \frac{\partial^{|\mathbf{b}|} f}{\partial (x^1)^{b_1}, \ \ldots, \ \partial (x^n)^{b_n}}. \tag{2.8}$$

Theorem 2.1.2. *Let d, d', α, and α' defined as in Definition 2.1.1 be given.*

(i) If $f \in \mathbf{A}^n_{\alpha,d}$, then $D^{\mathbf{b}} f \in \mathbf{A}^n_{\alpha,d}$;
(ii) If f is analytic in the domain $\mathscr{D}^n_{d'}$ and if for all $\bar{r}^* = \bar{r} + i\,\bar{\rho}$ in $\mathscr{D}^n_{d'}$ and constants $C > 0$, $m \geq 0$ and $\alpha'' \in (\alpha, \alpha')$, we have $f(\bar{r}^*) \leq C\, r^m \exp(-\alpha'' r)$, then $f \in \mathbf{A}^n_{\alpha,d}$.

Proof. Part (i): In the notation of Definition 2.1.1, taking $d'' \in (d, d')$ and $\alpha'' \in (\alpha, \alpha')$, we have for any $\bar{r}^* = (x^1, x^2, x^3) \in \mathscr{D}^n_{d''}$ and any $\varepsilon \in \min(0, d' - d'', (\alpha' - \alpha''))$, that

$$\frac{\partial f}{\partial x^j} = \frac{k\,!}{2\,\pi\, i} \int_{|\bar{r}^* - \bar{\rho}^*| = \varepsilon} \frac{f(\bar{\rho}^*)\, d\xi^j}{(\xi^j - x^j)^k}. \tag{2.9}$$

Hence by our assumption that for all $\bar{r}^* \in \mathscr{D}^n_{\alpha',d'}$ we have $|f(\bar{r}^*)| \leq C \exp(-\alpha' r)$, it follows that $|f(\bar{\rho}^*| \leq C \exp(-\alpha' \rho + \varepsilon) < C \exp(-\alpha'' \rho)$, and so

$$\left| \frac{\partial f}{\partial x^j} \right| \leq \frac{C}{2\,\pi\, \varepsilon^k} \exp(-\alpha'' r), \tag{2.10}$$

This proves Theorem 2.1.2 for the case of $f = f(\bar{r}^*)$ when taking one derivative with respect to x^j. The proof for the case of $D^{\mathbf{b}} f$ is similar, just by repeating the one-dimensional argument. The proof for the case of $f = f(\bar{r}^*, t)$ with $t \in [0, T]$ is also similar, and we omit it.

Part (ii). Let us select $\varepsilon > 0$, such that $\alpha'' - \varepsilon > \alpha$, and let us then select $R > 0$, such that if $r \geq R$, then $r^m \leq \exp(\varepsilon r)$. Then we have $C\, r^m\, e^{\alpha'' r} < C\, e^{\beta r}$, for $r \geq R$, where $\beta = \alpha'' - \varepsilon > \alpha$. We now select $C' > 0$ such that $C'\, e^{\varepsilon R} = C\, R^m$. Then $C'\, e^{\beta r} \leq C\, r^m\, e^{\alpha'' r}$ for $r \in (0, \infty)$.

■

2.2 Denseness of $\mathbf{A}^n_{\alpha,d}$ in $\mathbf{S}^n$

As we already mentioned, our preference is to work with the spaces $\mathbf{A}^n_{\alpha,d}$, not only for computing the solution of the N–S equations efficiently and accurately, but also for obtaining a simpler proof of existence of the solution to the equations. We first show in this section that the Sinc spaces are dense in the spaces $\mathbf{S}^n$. We then use this result in Sect. 2.2.2 to show that the spaces $\mathbf{A}^n_{\alpha,d}$ defined above are dense in the spaces $\mathbf{S}^n$.

Now, let $h > 0$, let $k \in \mathbb{Z}$, let $\xi \in \mathbb{C}$, let $\mathbf{k} \in \mathbb{Z}^n$, and let $\bar{r} \in \mathbb{R}^n$. Let us define the Sinc function $S(k,h)(\xi)$ in one dimension and a product $\mathscr{S}(\mathbf{k},h)(\bar{r}^*)$, in n dimensions, where now $\bar{r}^* \in \mathbb{C}^n$.

$$\begin{aligned} S(k,h)(x) &= \frac{\sin\left(\frac{\pi}{h}(x - k\,h)\right)}{\frac{\pi}{h}(x - k\,h)}, \\ \mathscr{S}(\mathbf{k},h)(\bar{r}^*) &= \prod_{j=1}^{n} S(k_j,h)(\xi^j), \quad \xi^j \in \mathbb{C}. \end{aligned} \tag{2.11}$$

2.2.1 Denseness of Sinc Approximation in $\mathbf{S}^n$

We shall prove the denseness of Sinc approximation only with respect to the "sup" norm; we omit the proofs of $\mathbf{L}^2$—approximation at this time, since such proofs follow almost verbatim from the $\mathbf{L}^\infty$ ones of this section. We thus prove only the following theorem in the remainder of this section.

Theorem 2.2.1. *Let $\bar{r} = (x^1, \ldots, x^n) \in \mathbb{R}^n$, let $g \in \mathbf{S}^n$, let $h > 0$, let N be a positive integer, and set*

$$g_{N,h}(\bar{r}) = \sum_{\mathbf{k} \in \mathbb{Z}_N^n} g(\mathbf{k}\,h)\,\mathscr{S}(\mathbf{k},h)(\bar{r}), \tag{2.12}$$

where

$$\mathbb{Z}_N^n = \{\mathbf{k} = (k^1, \ldots, k^n) \in \mathbb{Z}^n : -N \le k^j \le N,\ j = 1, \ldots, n\}. \tag{2.13}$$

Given any positive number ε, we can select $h > 0$ and N such that

$$\|g - g_{N,h}\|^\infty = \sup_{\bar{r} \in \mathbb{R}^n} |g(\bar{r}) - g_{N,h}(\bar{r})| < \varepsilon. \tag{2.14}$$

We split the proof of this theorem into the proofs of some lemmas.

We omit the proof of the following lemma since the (a)-Part of it is well known ([2], Theorem 1.2.1), and since the (b)-Part follows directly from the (a)-Part.

Lemma 2.2.1.

(a) If $(x,y) \in \mathbb{C} \times (-\pi/h, \pi/h)$, then

$$e^{ixy} = \sum_{j \in \mathbb{Z}} S(j,h)(x)\,e^{ijhy}. \tag{2.15}$$

(b) Let Q^n be defined by

$$Q^n = \left\{\bar{\rho} = (\xi^1, \ldots, \xi^n) \in \mathbb{R}^n : |\xi^j| < \pi/h,\ j = 1, \ldots, n\right\}. \tag{2.16}$$

Then we have the identity

$$\exp(i\,\bar{r} \cdot \bar{\rho}) = \sum_{\mathbf{k}\in\mathbb{Z}^n} \mathscr{S}(\mathbf{k}, h)(\bar{r})\ \exp(i\,j\,h\,\bar{\rho} \cdot \mathbf{k}) \tag{2.17}$$

for all $(\bar{r}, \bar{\rho}) \in \mathbb{C}^n \times Q^n$.

Remark 2.2.1. The function on the right-hand side of (2.15) can be extended as a function of y to the real line $\mathbb{R}$, where for arbitrary integer m it is a periodic copy of e^{ixy} on $(-\pi/h, \pi/h)$ to the interval $((2m-1)\,\pi/h, (2m+1)\,\pi/h)$. The infinite series on the right-hand side of (2.15) is discontinuous at each of the points $(2m+1)\,\pi/h$ where it takes on the value $\cos(\pi x)$. In particular, both functions, that on the left-hand side of (2.15) and that on the right-hand side, are bounded by 1 on $\mathbb{R} \times \mathbb{R}$. Similarly, both sides of (2.17) are identically equal on $\mathbb{C} \times Q^n$, and also the right-hand side of (2.17) has periodic extension to all of R^n, similar to that of (2.15) for the $n = 1$ case, and so both sides of (2.17) are bounded by 1 on $\mathbb{R}^n \times \mathbb{R}^n$.

Let $g_{N,h}$ be defined as in (2.12) and set

$$g_h(\bar{r}) = \lim_{N\to\infty} g_{N,h}(\bar{r}). \tag{2.18}$$

Lemma 2.2.2. *Let $g \in \mathbf{S}^n$, and let g_h be defined as in (2.18). Given $\varepsilon > 0$ there exists $h > 0$ such that*

$$\|g - g_h\|^\infty = \sup_{\bar{r}\in\mathbb{R}^n} |g(\bar{r}) - g_h(\bar{r})| < \frac{\varepsilon}{2}. \tag{2.19}$$

Proof. Let $\hat{g}$ denote the n-dimensional Fourier transform of g, i.e., with $\bar{r}$ and $\bar{\rho}$ in $\mathbb{R}^n$,

$$\hat{g}(\bar{r}) = \int_{\mathbb{R}^n} \exp(i\,\bar{r} \cdot \bar{\rho})\, g(\bar{\rho})\, d\bar{\rho}. \tag{2.20}$$

It then follows immediately, upon using (2.17) and applying the inverse of the Fourier transform formula of (2.20), that

$$\begin{aligned} &g(\bar{r}) - g_h(\bar{r}) \\ &= \frac{1}{(2\,\pi)^n} \int_{\mathbb{R}^n} \left(\exp(-i\,\bar{r} \cdot \bar{\rho}) - \sum_{\mathbf{k}\in\mathbb{Z}^n} \mathscr{S}(\mathbf{k}, h)(\bar{r})\ \exp(-i\,h\,\mathbf{k} \cdot \bar{\rho})\right) \cdot \\ &\quad \cdot \hat{g}(\bar{\rho})\, d\bar{\rho}. \end{aligned} \tag{2.21}$$

It is well known ([1], Corollary 3.2.1) that if $g \in \mathbf{S}^n$, then its Fourier transform, $\hat{g} \in \mathbf{S}^n$. Hence there exist constants $C > 0$ and $\beta > 0$, such that

$$|\hat{g}(\rho)| \le C\,(1+\rho)^{-n-\beta}. \tag{2.22}$$

Since the difference between each side of (2.21) is zero on Q^n and is bounded by 2 on $\mathbb{R}^n \setminus \overline{Q^n}$, and since $0 \le \rho^{n-1} < (1+\rho)^{n-1}$, we have, with Ω_n denoting[1] the "surface area" of the unit ball in $\mathbb{R}^n$, that

$$\begin{aligned}
\|g - g_h\|^\infty &\le \frac{1}{(2\,\pi)^n}\left|\int_{\mathbb{R}^n \setminus Q^n} g(\bar{\rho})\,d\bar{\rho}\right| \\
&\le \frac{2\,C}{(2\,\pi)^n}\int_{\mathbb{R}^n \setminus Q^n}(1+\rho)^{-n-\beta}\,d\bar{\rho} \\
&< \frac{2\,C}{(2\,\pi)^n}\int_{\pi/h}^{\infty}(1+\rho)^{-n-\beta}\,d\bar{\rho} \\
&= \frac{2\,C\,\Omega_n}{(2\,\pi)^n}\int_{\pi/h}^{\infty}\rho^{n-1}\,(1+\rho)^{-n-\beta}\,d\rho \\
&< \frac{2\,C\,\Omega_n}{(2\,\pi)^n}\int_{\pi/h}^{\infty}(1+\rho)^{-\beta-1}\,d\rho \\
&= \frac{2\,C\,\Omega_n}{(2\,\pi)^n\,\beta\,(1+\pi/h)^\beta}.
\end{aligned} \tag{2.23}$$

Under our assumption that $\beta > 0$, the right-hand side of (2.23) clearly approaches 0 as $h \to 0$. Hence, given any $\varepsilon > 0$, we can determine $h > 0$, such that $\|g(\bar{r}) - g_h(\bar{r})\|^\infty < \varepsilon/2$.

■

Lemma 2.2.3. *Let $a > 0$, $\gamma > 0$, set $w_\gamma(a,t) = \left(1 + (a^2+t^2)^{1/2}\right)^{-\gamma}$, assume that $h > 0$, and that N is a positive integer. Then,*

(i)

$$\sum_{|j|>N} w_{\gamma+1}(a, j\,h) \le \frac{2}{\gamma\,h}\,w_\gamma(a, N\,h). \tag{2.24}$$

[1] $\Omega_n = 2\,\pi^{n/2}/\Gamma(n/2)$.

(ii) If $\gamma \le 2(1+a)$, then

$$\sum_{j\in\mathbb{Z}} w_{\gamma+1}(a, jh) \le \frac{4}{h\gamma}\, w_\gamma(a,0). \tag{2.25}$$

Proof. Part (i). Note that $w_{\gamma+1}(a,t)$ is an even function of t which is monotonically decreasing on $(0,\infty)$. Since $|t|/(a^2+t^2)^{1/2} < 1$,

$$\begin{aligned}\sum_{|j|>N} w_{\gamma+1}(a,jh) &\le \frac{2}{h}\int_{Nh}^{\infty} w_{\gamma+1}(a,t)\,dt \\ &\le \int_{Nh}^{\infty} \frac{t}{(a^2+t^2)^{1/2}}\, w_{\gamma+1}(a,t)\,dt \\ &= \frac{2}{h\gamma}\, w_\gamma(a, Nh).\end{aligned} \tag{2.26}$$

Part (ii). Set

$$\begin{aligned} I(w) &= \frac{1}{h}\int_{\mathbb{R}} w_{\gamma+1}(a,t)\,dt \\ J(w) &= \sum_{j\in\mathbb{Z}} w_{\gamma+1}(a,jh). \end{aligned} \tag{2.27}$$

Then $J(w) \le I(w) + |J(w) - I(w)|$.

Let us first bound $I(w)$. Inserting the factor $|t|/(a^2+t^2)^{1/2} < 1$ into the integral in (2.27) yields

$$\begin{aligned} I(w) &\le \int_{\mathbb{R}} \frac{|t|}{(a^2+t^2)^{1/2}} \left(1+(a^2+t^2)^{1/2}\right)^{-1-\gamma} dt \\ &= 2\int_0^{\infty} \frac{t}{(a^2+t^2)^{1/2}} \left(1+(a^2+t^2)^{1/2}\right)^{-1-\gamma} dt \\ &= \frac{2}{\gamma}\, w_\gamma(a,0) = \frac{2}{\gamma\,(1+a)^\gamma}. \end{aligned} \tag{2.28}$$

Next, for any $\psi \in \mathbf{C}^1[0,h]$ we have the easily verified identity

$$\frac{h}{2}\,(\psi(0)+\psi(h)) = \int_0^h E_h(t)\psi'(t)\,dt, \tag{2.29}$$

where $E_h(t)$ is defined on $(0,h)$ by $E_h(t) = t - h/2$. We can extend the definition of $E_h(t)$ to all of $\mathbb{R}$ by setting $E_h(t) = E_h(t-mh)$ on $(mh,(m+1)h)$, with m

an arbitrary integer. We furthermore set $E_h(m\,h) = 0$, $m \in \mathbb{Z}$, and we then have $\sup_{t\in\mathbb{R}} |E_h(t)| \le h/2$.

Notice now that $w'_{\gamma+1}(t) > 0$ if $t < 0$, and $w'_{\gamma+1}(t) < 0$ if $t > 0$, and it follows, thus, from the definition of $J(w)$ given in (2.27) and our remarks following (2.29) that

$$|I(w) - J(w)| \le -2 \int_0^\infty w'_{\gamma+1}(a,t)dt = 2\, w_{\gamma+1}(a,0). \tag{2.30}$$

If we now add the right-hand sides of (2.28) and (2.30) we find, under the assumptions made in the statement of Lemma 2.2.2 (ii), that

$$\begin{aligned} J(w) \le I(w) + |I(w) - J(w)| &\le \frac{2}{h\,\gamma}\, w_\gamma(a,0) + 2\, w_{\gamma+1}(a,0) \\ &\le \frac{4}{h\,\gamma}\, w_\gamma(a,0). \end{aligned} \tag{2.31}$$

■

Lemma 2.2.4. *Let g_h and $g_{N,h}$ be defined as in (2.12). If $\mathbb{Z}^n_N$ is defined as in Theorem 2.2.1 and if h is selected as in Lemma 2.2.2, then we can select $N > 0$ such that*

$$\|g_h - g_{N,h}\|^\infty \le \frac{\varepsilon}{2}. \tag{2.32}$$

Proof. We shall again use our above definition of $w_{\gamma+1}$ i.e., taking $r = |\bar{r}|$, we set $w_{\gamma+1}(r) = (1+r)^{-\gamma-1}$, since it is now convenient to take $a = 0$ and $\gamma + 1 = n + \beta$, with $\beta > 0$. We can then write the difference $g_h - g_{N,h}$ as a telescoping sum,

$$\begin{aligned} \|g_h - g_{N,h}\|^\infty &= \sup_{\bar{r}\in\mathbb{R}^n} \left| \sum_{\mathbf{k}\in\mathbb{Z}^n\setminus\mathbb{Z}^n_N} g(h\,\mathbf{k})\, \mathscr{S}(\mathbf{k},h)(\bar{r}) \right| \\ &\le C \sum_{\mathbf{k}^n\in\mathbb{Z}^n\setminus\mathbb{Z}^n_N} w_{n+\beta}(h\,|\mathbf{k}^n|) \\ &\le C \sum_{\ell=1}^{n} \sigma^{(\ell)} \sum_{|k^\ell|>N} w_{n+\beta}(h\,|\mathbf{k}^{(\ell)}|). \end{aligned} \tag{2.33}$$

In this notation, the operator $\sigma^{(\ell)}$ denotes a product of $n-1$ sums, with $\ell - 1$ of them taken over $\mathbb{Z}^{\ell-1}_N$ and the other $n - \ell$ of them taken over $\mathbb{Z}^n \setminus \mathbb{Z}^\ell$, where $\mathbb{Z}^{n-1}_N$ is defined as in Theorem 2.2.1. More specifically, we can write $\sigma^\ell = \sum_{\mathbb{Z}^{\ell-1}_N} \sum_{\mathbb{Z}^n\setminus\mathbb{Z}^\ell}$ (with these sums being replaced by "1" when $\ell - 1 = 0$ and when $n - \ell = 0$).

Since $w_{n+\beta}(r) > 0$, we increase the right-hand side of (2.33) by replacing each truncated sum $\sum_{\mathbf{k}^\ell \in \mathbb{Z}^\ell_N}$ with $\sum_{\mathbf{k}^\ell \in \mathbb{Z}^\ell}$, $\ell = 1, \ldots, n-1$. Similarly (note, at the symmetry and positivity of $w_{n+\beta}(r)$ enable us to replace each double sum $\sum_{k^1 \in \mathbb{Z}} \sum_{|k_\ell| > N}$ occurring in the sum line of $\sigma^{(\ell)}$ with $\sum_{\ell \in \mathbb{Z}} \sum_{|k^1| > N}$, for $\ell = 2, \ldots, n$. Upon doing this, and taking $\mathbf{k}^{n-1} = \{(k^2, \ldots, k^n) \in \mathbb{Z}^{n-1}\}$, each ℓth row of sums becomes the same, i.e., we get n such sums, so that

$$\|g_h - g_{N,h}\|^\infty \le C\, n \sum_{\mathbf{k}^{n-1} \in \mathbb{Z}^{n-1}} \sum_{|k^1| > N} w_{n+\beta}(h\,|k^1, \mathbf{k}^{n-1}|). \tag{2.34}$$

Applying Lemma 2.2.3 (i) to bound the sum with respect to k^1 on the right of (2.34), we get

$$\|g_h - g_{N,h}\|^\infty \le \frac{2\, C\, n}{(n+\beta-1)\, h} \sum_{\mathbf{k}^{n-1} \in \mathbb{Z}^{n-1}} w_{n+\beta-1}(h\,|N, \mathbf{k}^{n-1}|). \tag{2.35}$$

We next successively apply Lemma 2.2.3 (ii) to each single sum in the product of $n-1$ sums of (2.35), one at a time, under the assumption that $(1 + N\,h) > \beta$. After applying Lemma 2.2.3 (ii) to the first sum, the resulting (2.35) becomes

$$\begin{aligned} &\|g_h - g_{N,h}\|^\infty \\ &\le \frac{4\,(2\,C\,n)}{(n+\beta-1)(n+\beta-2)\, h^2} \sum_{\mathbf{k}^{n-2} \in \mathbb{Z}^{n-2}} w_{n+\beta-2}(h\,|N, 0, \mathbf{k}^{n-2}|). \end{aligned} \tag{2.36}$$

Hence repeating this process another $n-2$ times, we get our final result,

$$\|g_h - g_{N,h}\|^\infty \le \frac{4^{n-1}\,(2\,C\,n)}{h^n\,(\beta)_n\,(1+N\,h)^\beta}, \tag{2.37}$$

in which $(\beta)_n = \beta\,(\beta+1), \cdots, (\beta+n-1)$.

Since h is fixed, it is clear that the right-hand side of (2.37) approaches zero as $N \to \infty$, and we can thus select N so that $2\,(1 + N\,h) > \beta$, and also, so that the right-hand side of (2.37) is less than $\varepsilon/2$.

■

By combining the results of Lemmas 2.2.2 and 2.2.4 we get $\|g - g_{N,h}\|^\infty \le \|g - g_h\|^\infty + \|g_h - g_{N,h}\|^\infty < \varepsilon/2 + \varepsilon/2 = \varepsilon$.

This completes the proof of Theorem 2.2.1. ■

2.2.2 Denseness of the Space $\mathbf{A}^n_{\alpha,d}$ in $\mathbf{S}^n$

Let $\bar{r}$ and $\bar{\rho}$ belong to $\mathbb{Z}^n_N$, let $\mathscr{S}(\mathbf{k},h)(\bar{r})$ be defined as in the above subsection, let $g \in \mathbf{S}^n$, let $\mathbb{Z}^n_N$ be defined as in (2.13), and for arbitrary fixed $\bar{\rho} \in \mathbb{R}^n$, let us consider the Sinc approximation of $\kappa(g,\bar{\rho},\bar{r}) \equiv \exp\left(-|\bar{r}-\bar{\rho}|^2\right)\, g(\bar{r})$, i.e., let us examining the difference

$$E_{N,h}(\kappa) = \sup_{\bar{r}\in\mathbb{R}^n} \left| \kappa(g,\bar{\rho},\bar{r}) - \sum_{\mathbf{k}\in\mathbb{Z}^n_N} \kappa(g,\bar{r},h\,\mathbf{k})\, \mathscr{S}(\mathbf{k},h)(\bar{r}) \right|. \tag{2.38}$$

Evidently, given any $\bar{\rho} \in \mathbb{R}^n$, $\kappa \in \mathbf{S}^n$ whenever $g \in \mathbf{S}^n$. Hence, it follows that Theorem 2.2.1 applies to this function κ uniformly, for $\bar{\rho} \in \mathbb{R}^n$. We thus take $\bar{\rho} = \bar{r}$ in (2.38), to get $\kappa(g,\bar{r},\bar{r}) = g(\bar{r})$ enabling us to state the following result:

Theorem 2.2.2. *Given any $g \in \mathbf{S}^n$, and given any positive number ε, there exists a positive integer N, constants $c_{\mathbf{k}}$, and functions $w_{\mathbf{k}}, \mathbf{k} \in \mathbb{Z}^n_N$ with each of the $w_{\mathbf{k}}$ belonging to $\mathbf{A}^n_{\alpha,d}$, such that*

$$\left\| g - \sum_{\mathbf{k}\in\mathbb{Z}^n_N} c_{\mathbf{k}}\, w_{\mathbf{k}} \right\|^{\infty} < \varepsilon. \tag{2.39}$$

Proof. The above description of κ taken together with Theorem 2.2.1 enables us to state that given any $\varepsilon > 0$, there exists a number $h > 0$ and a positive integer N such that

$$\sup_{\bar{r}\in\mathbb{R}^n} \left| g(\bar{r}) - \sum_{\mathbf{k}\in\mathbb{Z}^n_N} g(h\,\mathbf{k})\, \exp\left(-|\bar{r}-h\,\mathbf{k}|^2\right)\, \mathscr{S}(\mathbf{k},h)(\bar{r}) \right| < \varepsilon. \tag{2.40}$$

The proof is completed upon noting that each of the functions

$$w_{\mathbf{k}}(\bar{r}) = \exp\left(-|\bar{r}-h\,\mathbf{k}|^2\right)\, \mathscr{S}(\mathbf{k},h)(\bar{r}) \tag{2.41}$$

belongs to $\mathbf{A}^n_{\alpha,d}$ for all positive numbers α and d.
■

References

1. R. Narasimhan, *Analysis on Real and Complex Manifolds* (North-Holland, Amsterdam, 1968)
2. F. Stenger, *Handbook of Sinc Numerical Methods*. A 470-page Tutorial & Matlab Package (CRC Press, Boca Raton, 2011)
3. E.C. Titchmarsh, *Introduction to the Theory of Fourier Integrals* (Oxford University Press, Oxford, 1937)

Chapter 3
Spaces of Solution of the N–S Equations

Abstract In this chapter we prove that if each component of the vector $\mathbf{u}$ on the right-hand side of (1.11) is divergence-free and belongs to the space of functions $\mathbf{A}^3_{\alpha,d,T}$, then the same is true of the operation $\mathbf{N}\,\mathbf{u}$ on the right-hand side of (1.11). We also derive some bilinear form expressions for the operation $\mathbf{N}\,\mathbf{u}$ thus paving the way for our proof of existence of a solution to (1.11). We then give precise conditions for convergence of successive approximations to the solution of (1.11) based on the contraction mapping principle.

3.1 Banach Spaces

In this section we introduce Banach spaces $\mathscr{B}$ and $\mathscr{B}_T$ which are convenient for studying the operator equations (1.23)–(1.24).

Thus, for given $\mathbf{u} = (u^1, u^2, u^3)$, we use the notations $\|\cdot\|$ for our 2-norm defined as in (2.4), to define our norms of the vector $\mathbf{u}$. Letting $\|\cdot\|$ and $\|\cdot\|_T$ be defined as in (2.4)–(2.5), we set

$$|||\mathbf{u}||| = \left(\sum_{j=1}^{3} \|u^j\|^2\right)^{1/2}, \quad \text{and}$$
$$|||\mathbf{u}|||_T = \left(\sum_{j=1}^{3} \|(u^j\|_T^2\right)^{1/2} \tag{3.1}$$

depending on whether $\mathbf{u}$ is independent or dependent on T respectively. Note also that $|||\hat{\mathbf{u}}||| = \left((2\,\pi)^3\right)^{1/2} |||\mathbf{u}|||$ and similarly for $|||\hat{\mathbf{u}}|||_T$.

It is convenient at this point to also introduce a Banach space of functions.

Definition 3.1.1.

(i) Let α and d denote arbitrary positive numbers. Let $\mathscr{B}$ denote the Banach space of vector functions $\mathbf{w} = (w^1, w^2, w^3)$, with $\mathbf{w}$ divergence-free, with each w^j belonging to $\mathbf{A}^3_{\alpha,d}$, for $j = 1, 2, 3$, and with $|||\mathbf{w}||| < \infty$.

© Springer International Publishing AG 2016

F. Stenger et al., *Navier–Stokes Equations on* $\mathbb{R}^3 \times [0, T]$,
DOI 10.1007/978-3-319-27526-0_3

(ii) Let $\mathscr{B}_T$ denote the family of all vector functions $\mathbf{u} = (u^1, u^2, u^3)$ with $\mathbf{u}(\cdot, t) \in \mathscr{B}$ for all fixed $t \in (0, T)$ with $u^j \in \mathbf{A}^3_{\alpha,d,T}$ and with $|||\mathbf{u}|||_T < \infty$.

We shall then prove, in essence, that if $\mathbf{u} \in \mathscr{B}_T$, then $\mathbf{N}\,\mathbf{u} \in \mathscr{B}_T$. This statement will be made more precise in what follows.

The following lemma plays an important role in the proofs which follow.

Lemma 3.1.1. *Let δ be a positive number, let $\mathscr{D}^3_\delta$ be defined as in Definition 2.1.1, and let $\bar{\Lambda}^* \in \mathscr{D}^3_\delta$. If $0 \le t < T < \infty$, then the Green's function $\mathscr{G}$ given in (1.6) has the Fourier transform*

$$\hat{\mathscr{G}}(\bar{\Lambda}^*, t) = \int_{\mathbb{R}^3} \exp(i\,\bar{\Lambda}^* \cdot \bar{r})\,\mathscr{G}(\bar{r}, t)\,d\bar{r} = \exp\left(-(\Lambda^*)^2\,\varepsilon\,t\right). \tag{3.2}$$

(a) The function $\hat{\mathscr{G}}(\cdot, t)$ is analytic and bounded in $\mathscr{D}^3_\delta$.
(b) Furthermore,

$$\lim_{T\to\infty} \sup_{(\bar{\Lambda}^*, t)\in\mathscr{D}^3_\delta\times[0,T]} |\hat{G}(\bar{\Lambda}^*, t)| = \infty. \tag{3.3}$$

In particular, $\hat{G}(\bar{\Lambda}^*, t)$ is unbounded in the region $\{\mathbb{C}^3 \times [0, T] : |\Im\Lambda^*| > |\Re\Lambda^*|\}$ as $T \to \infty$.

Proof. Let us observe by (3.2), that if $(\bar{\Lambda}^*, t) \in \mathbb{C}^3 \times [0, T]$, then

$$\sup_{\Lambda^*\in\,\mathbb{C}^3, t\in[0,T]} |\hat{G}(\bar{\Lambda}^*, t)| = \exp\left(|\Im\Lambda^*|^2\,T\right). \tag{3.4}$$

The statement of Lemma 3.1.1 follows by inspection of this identity.

■

3.2 The Vector $\mathbf{v} = \mathbf{G}\,\mathbf{u}^0$

Let us begin by bounding the norm of $\mathbf{v} = \mathbf{G}\,\mathbf{u}^0$ with $\mathbf{u}^0$ defined as in (1.3), and with $|||\mathbf{w}|||$ defined as in (3.1). We shall also show that if $\mathbf{u}^0$ is divergence-free, then $\mathbf{v} \in \mathscr{B}_T$, where $\mathbf{v}$ is defined as in (1.12).

Theorem 3.2.1. *If $0 < T < \infty$, and if $\mathbf{u}^0 \in \mathscr{B}$, then $\mathbf{v} \in \mathscr{B}_T$ for all finite $T > 0$. Moreover,*

$$|||\mathbf{v}|||_T = |||\mathbf{u}^0|||. \tag{3.5}$$

Proof. Taking the Fourier transform of each side of the first equation in (1.12), we get

$$\begin{aligned}\hat{\mathbf{v}}(\bar{\Lambda}, t) &= \int_{\mathbb{R}^3} \exp(i\,\bar{r}\,\cdot\,\bar{\Lambda}^*)\,\mathbf{u}^0(\bar{r})\,d\bar{r} \\ &= \widehat{\mathbf{u}^0}(\bar{\Lambda}^*)\,\exp(-\varepsilon\,(\Lambda^*)^2\,t).\end{aligned} \tag{3.6}$$

where $\bar{\Lambda}^* = \bar{\Lambda} + i\,\bar{\rho}$, with $\bar{\Lambda}$ and $\bar{\rho}$ belonging to $\mathbb{R}^3$ and with $\rho = |\bar{\rho}| \le d'$.

Since $\mathbf{u}^0 \in \mathscr{B}$, each component of $\mathbf{u}^0$ belongs to $\mathbf{A}^3_{\alpha,d}$ so that by Theorem 2.1.1, each component of $\widehat{\mathbf{u}^0}$ belongs to $\mathbf{A}_{d,\alpha}$, and hence each of these components is analytic in $\mathscr{D}^3_{\alpha'}$. Inspection of (3.6) shows that $\hat{\mathbf{v}} \in \mathbf{A}_{d,\alpha,T}$ if the function $\exp(-\varepsilon\,\Lambda^2\,t)$ is analytic and bounded in $\mathscr{D}^3_{\alpha'}$ for each fixed $t \in [0, T]$ and in addition, if this same function is also an analytic and bounded function of $t \in \mathscr{D}_{d',T}$ for each fixed $\bar{\Lambda} \in \mathbb{R}^3$.

Hence each component of $\mathbf{v}$ belongs to $\mathbf{A}^3_{\alpha,d,T}$.

Finally, note that $\mathbf{v}$ is divergence-free as a consequence of Lemma 1.3.1. The norm-bound (3.5) is a consequence of $\exp(-\varepsilon\Lambda^2\,t) \le 1$ for $\bar{\Lambda} \in \mathbb{R}^3$, yielding $|||\hat{\mathbf{v}}||| < |||\hat{\mathbf{u}}^0|||$ for $t \in (0, T]$ together with $|||\hat{\mathbf{v}}||| = |||\hat{\mathbf{u}}^0|||$ for $t = 0$, and (3.1).We thus arrive at (3.5) and at our conclusion that $\mathbf{v} \in \mathscr{B}_T$.

■

3.3 The Operation $\mathbf{G}_{x^\ell}\mathbf{w}$

Let w be defined on $\mathbb{R}^3 \times (0, \infty)$, let $\mathscr{G}(\bar{r}, t)$ be defined as in (1.6), and let us define the operation $\mathbf{G}_{x^\ell}\,w$ on $\mathbb{R}^3 \times (0, \infty)$ by

$$\mathbf{G}_{x^\ell}\,w(\bar{r}) = \int_0^t \int_{\mathbb{R}^3} \mathscr{G}_{x^\ell}(\bar{r} - \bar{\rho}, t - t')\,w(\bar{\rho}, t')\,d\bar{\rho}\,dt'. \tag{3.7}$$

Theorem 3.3.1. *If either $w \in \mathbf{A}^3_{2\,\alpha,d}$ or else if $w \in \mathbf{A}^3_{2\,\alpha,d,T}$ for $0 < T < \infty$, then $\mathscr{G}_{x^\ell}(w) \in \mathbf{A}^3_{2\,\alpha,d,T}$ for all such T. We furthermore have*

$$\begin{aligned}&\|\mathbf{G}_{x^\ell}(w)\|_T \le 2\left(\frac{T}{\pi\,\varepsilon}\right)^{1/2}\|w\|, \quad w \in \mathbf{A}^3_{\alpha,d}, \text{ or if } w \in \mathbf{A}^3_{2\,\alpha,d}, \\ &\|\mathbf{G}_{x^\ell}(w)\|_T \le 2\left(\frac{T}{\pi\,\varepsilon}\right)^{1/2}\|w\|_T, \quad w \in \mathbf{A}^3_{\alpha,d,T}.\end{aligned} \tag{3.8}$$

Proof. We have

$$(\mathbf{G}_{x^\ell}\,w)(\bar{r}, t) = \int_0^t \int_{\mathbb{R}^3} \mathscr{G}_{x^j}(\bar{r} - \bar{r}', t - t')\,w(\bar{r}', t')\,d\bar{r}'\,dt'. \tag{3.9}$$

Let $\alpha'' \in (\alpha, \alpha')$, and let $d'' \in (d, d')$. The Fourier transform of $\mathbf{G}_{x^j} w$ is given for $\bar{\Lambda}^* \in \mathscr{D}^3_{2\alpha'',T}$ by

$$\widehat{\mathbf{G}_{x^j} w}(\bar{\Lambda}^*, t) = \int_0^t i\,(\lambda^j)^* \exp\left(-(\Lambda^*)^2\,(t-t')\right)\,\hat{w}(\bar{\Lambda}^*, t')\,dt'. \tag{3.10}$$

Here $(\lambda^j)^*$ is the complex-valued jth component of $\Lambda^* \in \mathbb{C}^3$, with $\Lambda \in \mathbb{R}$ and with $|\Im(\Lambda^j)^*| \le 2\,\alpha''$. Inspection of this equation in view of Lemma 3.1.1 enables us to conclude the following:

(i) Since $w \in \mathbf{A}^3_{2\,\alpha,d,T}$, $\hat{w} \in \mathbf{A}^3_{d,2\,\alpha,T}$, by Theorem 2.1.1 , so that, by Theorem 3.3.1, $\widehat{\mathbf{G}_{x^j} w} \in \mathbf{A}^3_{d,2\,\alpha,T}$.

(ii) $\widehat{\mathbf{G}_{x^\ell} w}(\bar{\Lambda}^*, \cdot)$ is analytic and uniformly bounded in $\mathscr{D}_{2\,\alpha'',T}$ for each fixed $\Lambda^* = \bar{\Lambda} + i\,\bar{\rho} \in \mathscr{D}_{d'',2\,\alpha''}$.

By (i) and (ii), $\mathbf{G}_{x^\ell}(w) \in \mathbf{A}^3_{2\,\alpha,d,T}$.

Next, the identity

$$\max_{t\in[0,T]} \int_0^t \int_{\mathbb{R}^3} \left|\mathscr{G}_{x^\ell}(\bar{r}, t-t')\right| d\bar{r}\,dt' = 2\left(\frac{T}{\pi\,\varepsilon}\right)^{1/2}, \quad \ell = 1,2,3, \tag{3.11}$$

yields (3.8)

■

3.4 The Operation $p = \mathbf{G_0}\,g$

Let p, g, and $\mathscr{G}_0$ be defined as in Theorem 1.4.1, so that we can also write

$$p(\bar{r}, t) = (\mathbf{G}_0\,g)(\bar{r}, t) = \int_{\bar{\rho}\in\mathbb{R}^3} G_0(\bar{r}-\bar{\rho})\,g(\bar{\rho}, t)\,d\bar{\rho}. \tag{3.12}$$

Our aim is to show that if $g \in \mathbf{A}^3_{2\,\alpha,d,T}$, then $\mathbf{G}_0\,g \in \mathbf{A}^3_{2\,\alpha,d,T}$. This requires showing:

(i) *Analyticity*. If g is analytic in $\mathscr{D}^3_{d',T}$, then $\mathbf{G}_0\,g$ is analytic in $\mathscr{D}^3_{d',T}$,; and

(ii) *Asymptotic relation*. That if $g(\bar{r}^*, t) = \mathscr{O}\,(\exp(-2\,\alpha\,r))$ as $r = |\bar{r}| \to \infty$ for all $t \in [0,T]$, then $(\mathbf{G}_0\,g)(\bar{r}^*, t) = \mathscr{O}\,(\exp(-2\,\alpha\,r))$ as $|\bar{r}| \to \infty$, for all $t \in [0,T]$.

Theorem 3.4.1. *Let $\mathscr{G}_0$, g, and p be defined as in Theorem 1.4.1, let $p = \mathbf{G}_0\,g$ denote the right-hand side of (3.12). If $g \in \mathbf{A}_{2\,\alpha,d,T}$, then $\mathbf{G}_0\,g \in \mathbf{A}_{2\,\alpha,d,T}$.*

We shall prove Theorem 3.4.1 via the proofs of several lemmas. Except for the analyticity part, we shall omit the dependence on t in these proofs, since all of the results hold uniformly for $t \in [0,T]$. We shall also omit mentioning, except when necessary, that bounds such as $|z(\bar{r}^*)| \le C\,\exp(-2\,\alpha\,r)$ hold for all $\bar{r}^* = \bar{r} + i\,\Im\bar{r}' \in \mathscr{D}^3_d$.

Lemma 3.4.1. *Let* $\mathbf{u} = (u^1, u^2, u^3)$ *be divergence-free, let* g *be defined as in Theorem 1.4.1, and define* f *by*

$$f(\mathbf{u}) = \sum_{j=1}^{3} \sum_{k=1}^{3} D^{j,k}(u^j\, u^k). \tag{3.13}$$

Then $f = g$.

Proof. The proof of this lemma is a part of the derivation of Eq. (1.14) and is omitted.

■

Lemma 3.4.2. *Let* f *be defined as in (3.13). Then*

$$\int_{\mathbb{R}^3} f(\bar{r})\, d\bar{r} = 0. \tag{3.14}$$

Proof. The statement of the lemma follows readily using integration by parts in each of the integral $\int_{\mathbb{R}^3} D^{j,k}\,(u^j\, u^k) d\bar{r}$.

■

Lemma 3.4.3. *Let* p *be defined as in (3.12). If the conditions of Theorem 3.4.1 are satisfied, so that* u^j *and* u^k *are analytic in* $\mathscr{D}^3_{d',T}$, *where, with* d *as in Theorem 3.4.1 and with* $0 < d < d'$, *then* p *is analytic in* $\mathscr{D}^3_{d',T}$.

Proof. Let u^j and u^k belong to $\mathbf{A}^3_{\alpha,d',T}$, so that $z \equiv D^{j,k}(u^j\, u^k) \in \mathbf{A}^3_{2\,\alpha,d',T}$, with z being one of the terms in the definition (3.13) of f. Upon replacing $\bar{\rho}$ in (3.12) with $\bar{R}^* - \bar{\rho}$, we get

$$p(\bar{R}^*) = \frac{1}{4\,\pi} \int_{\mathbb{R}^3} \frac{f(\bar{R}^* - \bar{\rho})}{\rho}\, d\bar{\rho}. \tag{3.15}$$

Since each component of f is analytic in $\mathscr{D}^3_{d',T}$, it follows by inspection of (3.15) that p is also analytic in $\mathscr{D}^3_{d',T}$.

■

We shall require the following result involving potentials.

Lemma 3.4.4. *Let* $\rho = |\rho|\, \Omega$, *so that* $d\bar{\rho} = \rho^2\, d\Omega(\bar{\rho})\, d\rho$, *where* $\Omega \in \mathbb{R}^3$ *and* $|\Omega| = 1$. *Then,*

$$\frac{1}{4\,\pi} \int_{|\Omega(\bar{\rho})|=1} \frac{d\Omega(\bar{\rho})}{|\bar{r} - \bar{\rho}|} = \min(1/\rho, 1/r). \tag{3.16}$$

Proof. Let m and k denote nonnegative integers. By expanding both sides of (1.22) in powers of $\Lambda\, r$ we get

$$\int_{|\Omega(\bar\rho)|=1} (\bar r\cdot\,\bar\rho)^n\, d\Omega(\bar\rho)^m = 4\,\pi \begin{cases} 0 \text{ if } m = 2\,k+1 \\ \dfrac{(r\,\rho)^{2\,k}}{2\,k} \text{ if } n = 2\,k. \end{cases} \tag{3.17}$$

Next, expanding $|\bar r - \bar\rho|^{-1} = \left(r^2+\rho^2-2\,\bar r\cdot\,\bar\rho\right)^{-1/2}$ in powers of $t = 2\,\bar r\cdot\,\bar\rho\,/(r^2+\rho^2)$, we get

$$\begin{aligned} Q(r,\rho) &\equiv (r^2+\rho^2)^{-1/2} \sum_{n=0}^{\infty} \frac{(1/2)_{2n}}{(2n)\,!} \left(\frac{2\,r\,\rho}{r^2+\rho^2}\right)^{2n} \\ &= (r^2+\rho^2)^{-1/2} \sum_{n=0}^{\infty} \frac{(1/4)_n\,(3/4)_n}{(3/2)_n\,n!} \left(\frac{2\,r\,\rho}{r^2+\rho^2}\right)^{2n} \\ &= (r^2+\rho^2)^{-1/2}\, F\left(1/4, 3/4;\, 3/2; \left(\frac{2\,r\,\rho}{r^2+\rho^2}\right)^2\right). \end{aligned} \tag{3.18}$$

In the last equation of (3.18) F denotes the usual *hypergeometric function*,

$$F(a,b;c;\,z) = \sum_{n=0}^{\infty} \frac{(a)_n\,(b)_n}{(c)_n\,n!}\, z^n \tag{3.19}$$

where this series representation is defined and converges for all $|z| < 1$ whenever, e.g., $a > -1$, $b > -1$, and $c > -1$, and where, for any $\alpha \in \mathbb{C}$, and n a nonnegative integer,

$$(\alpha)_n = \begin{cases} 1 \text{ if } n = 0 \\ \alpha\,(\alpha+1)\cdots(\alpha+n-1) \text{ if } n > 0. \end{cases} \tag{3.20}$$

We have also used the identity $(2\,\alpha)_{2\,n} = 2^{2\,n}\,(\alpha)_n\,(\alpha+1/2)_n$ in (3.18).

Next, we use the *quadratic transformation identity*—see e.g., [1],

$$F\left(a,b;\,2\,b;\,\frac{4\,z}{(1+z)^2}\right) = (1+z)^{2\,a}\,F(a,a+1/2-b;b+1/2;\,z^2), \tag{3.21}$$

where it is assumed that $|z| < 1$ on the right-hand side of (3.21). Upon substituting the numerical values of a, b, and c of (3.18) into (3.21), we get

$$F\left(1/4, 3/4;\ 3/2;\ \frac{4\,z}{(1+z)^2}\right) = (1+z)^{1/2}\, F\left(1/4, 0;\ 5/4;\ z^2\right) \\ = (1+z)^{1/2}, \quad |z| < 1, \tag{3.22}$$

where the second line of (3.22) obtains as a result of $F(1/4, 0;\ 5/4;\ z^2) = 1$.

At this point we take $r \in (0, \infty)$, and we consider two cases of (3.22), $0 < \rho < r$ and $r < \rho < \infty$. In the first case, taking $z = (\rho/r)^2$, we get $Q(r, \rho) = 1/r$, whereas in the second, taking $z = (r/\rho)^2$ we get $Q(r, \rho) = 1/\rho$. (Note: We get $4z/(1+z)^2 = 4(r\,\rho)^2/((r^2+\rho)^2)^2$ for each of these cases.)

■

We next proceed to prove the *asymptotic relation* stated at the outset of this section.

Let us split $\mathbb{R}^3$ into the two regions, $B_i = \{\bar{\rho} \in \mathbb{R}^3 : |\bar{\rho}| < r\}$ and $B_o = \mathbb{R}^3 \setminus B_i$. Let us also split the integral defining $p = \mathbf{G}_0 f$ into a sum of two integrals, p_i and p_o, i.e., $p = p_i + p_o$, where

$$p_i(\bar{r}) = \int_{B_i} \frac{f(\bar{\rho})}{4\,\pi\,|\bar{r} - \bar{\rho}|}\, d\bar{\rho} \\ p_o(\bar{r}) = \int_{B_o} \frac{f(\bar{\rho})}{4\,\pi\,|\bar{r} - \bar{\rho}|}\, d\bar{\rho}. \tag{3.23}$$

Lemma 3.4.5. *There exists a constant C such that the function $p = p_i + p_o$ satisfies the inequality*

$$|p(\bar{r})| = |p_i(\bar{r}) + p_o(\bar{r})| \le C\, \exp(-2\,\alpha\, r). \tag{3.24}$$

Proof. We shall assume without loss of generality that f is real-valued on $\mathbb{R}^3$.

We split the proof of this lemma into two parts: Part I for p_i, and Part II for p_o.

Proof. Part I: Given f, we set $f = f_+ + f_-$, where $f_+(\bar{\rho}) = f(\bar{\rho})$ if $f(\bar{\rho}) > 0$, and 0 if $f(\bar{\rho}) < 0$, and where $f_-(\bar{\rho}) = f(\bar{\rho}) - f_+(\bar{\rho})$. For $\bar{\rho}$ and $\bar{r} \in B_i$, the function $\mathscr{G}_0(\bar{r} - \bar{\rho})$ defined as in Theorem 1.4.1 is positive and varies over the range $1/(8\,\pi\,r) \le \mathscr{G}(\bar{\rho}) < \infty$. Hence, setting

$$\tilde{f}_\pm = \int_{B_i} f_\pm(\bar{\rho})\, d\bar{\rho}, \tag{3.25}$$

and applying the mean value theorem, we get

$$\begin{aligned} p_{i,\pm}(\bar{r}) &\equiv \int_{|\bar{\rho}|<r} \frac{f_{\pm}(\bar{\rho})}{4\,\pi\,|\bar{\rho}-\bar{r}|}\,d\bar{\rho} \\ &= G_{\pm}\tilde{f}_{\pm}, \end{aligned} \tag{3.26}$$

where $G_{\pm}$ are values of $\mathscr{G}(\bar{r}-\bar{\rho}) \in [1/(8\,\pi\,r), \infty)$. Hence

$$p_i(\bar{r}) = G_+\tilde{f}_+ + G_-\tilde{f}_-. \tag{3.27}$$

Upon keeping the quantities $\tilde{f}_{\pm}$ and $G_+\tilde{f}_+ + G_-\tilde{f}_-$ fixed, we can decrease the larger of the two of G_+ and G_- and increase the smaller, until the two values become the same value, call it G^*. Thus, e.g., G_+ is replaced with $G^* = G_+ - h$ and G_- with $G^* = G_- + k$, this being the first of two equations for unknowns h and k. The second equation obtains from the statement that $G_+\tilde{f}_+ + G_-\tilde{f}_-$ is unchanged. These two equations become, in matrix form,

$$\begin{pmatrix} \tilde{f}_+ & -\tilde{f}_- \\ 1 & 1 \end{pmatrix} \begin{pmatrix} h \\ k \end{pmatrix} = \begin{pmatrix} 0 \\ G_+ - G_- \end{pmatrix} \tag{3.28}$$

Now, if $\tilde{f}_+ + \tilde{f}_- \neq 0$, then the determinant of the matrix in (3.28) does not vanish, and this system then has the unique solution (h, k), yielding $G^* = (\tilde{f}_+ G_+ + \tilde{f}_- G_-)/(\tilde{f}_+ + \tilde{f}_-)$. On the other hand, if $\tilde{f}_+ + \tilde{f}_- = 0$, then we must have $G_+ - G_- = 0$, i.e., $G^* = G_+ = G_-$.

By Lemma 3.4.2 we thus always have

$$\begin{aligned} p_i(\bar{r}) &= G^*\,\left(\tilde{f}_+ + \tilde{f}_-\right) \\ &= G^* \int_{B_i} f(\bar{\rho})\,d\bar{\rho} \\ &= -G^* \int_{B_o} f(\bar{\rho})\,d\bar{\rho} \\ &= G^*\mathscr{O}(\exp(-2\,\alpha'\,r)), \quad r \to \infty. \end{aligned} \tag{3.29}$$

Furthermore, if f is complex-valued, then we can treat the imaginary part of f in the same way as above.

We also need to bound the coefficient G^* in (3.29). Recall that G^* is between the largest and smallest of G_+ and G_-. These three numbers, $G_{\pm}$ and G^* are bounded by the norm $\|\mathbf{G}_0\|$ of the operator equation $p = \mathbf{G}_0 f$, i.e., by using the $\mathbf{L}^1$ norm, $\|\cdot\|_1 = \int_{B_i} |f(\bar{\rho})|\,d\bar{\rho}$, we get $|p_i(\bar{r})| \leq \|\mathbf{G}_0\|\,\|f\|_1$. However, since we are not dealing with the whole space $\mathbf{L}^1(B_i)$, but rather with the subspace $\mathbf{A}_{2\,\alpha,d}$ restricted to $\mathbf{L}^1(B_i)$, we can confine ourselves to bounding this norm in this space functions. Since each of these functions is bounded by $C\,\exp(-2\,\alpha'\,r)$, with C and $\alpha' > \alpha$ denoting positive numbers, every member w of this class can be written as a product of a constant C times $U(\bar{\rho})\,\exp(-2\,\alpha'\,\rho)$, where U is bounded by 1 in B_i.

Thus, as a result of $\sup_{|U(\bar\rho)|\le 1} |U(\bar\rho)| = 1$,

$$\begin{aligned} \|w\| &= \int_{B_i} |w(\bar\rho)|\, d\bar\rho \\ &= \sup_{|U(\bar\rho)|\le 1} \int_{B_i} \exp(-2\,\alpha'\,\rho)\, |U(\bar\rho)|\, d\bar\rho \\ &= \int_{B_i} \exp(-2\,\alpha'\,\rho)\, d\bar\rho. \end{aligned} \tag{3.30}$$

Next, for $\mathbf{G}_0\, w$, we have, as a consequence of Lemma 3.4.4 that

$$\begin{aligned} |\,(\mathbf{G}_0\, w)\,(\bar r)| &\le \left| \int_{B_i} \mathscr{G}_0(\bar r - \bar\rho)\, \exp(-2\,\alpha'\,\rho)\, U(\bar\rho)\, d\bar\rho \right| \\ &\le \sup_{|U(\bar\rho)|\le 1} \int_{B_i} \mathscr{G}_0(\bar r - \bar\rho)\, e^{-2\,\alpha'\,\rho}\, |U(\bar\rho|\, d\bar\rho \\ &= \int_0^r \int_{|\Omega|=1} \mathscr{G}_0(\bar r - \bar\rho)\, d\Omega\, \exp(-2\,\alpha'\,\rho)\, \rho^2\, d\rho \\ &= \frac{1}{r} \int_0^r \exp(-2\,\alpha'\,\rho)\, \rho^2\, d\rho = (1 - e^{-2\alpha' r})/(4\alpha'^3 r)\, \|w\|. \end{aligned} \tag{3.31}$$

That is, from $|\mathbf{G}_0\, w| \le \|\mathbf{G}_0\|\, \|w\|$, we have $\|\mathbf{G}_0\| \le (1 - e^{-2\alpha' r})/(4\alpha'^3 r)$.

It follows therefore, from Eq. (3.30) and Theorem 2.1.1, that there exists a constant C such that

$$|p_i(\bar r)| \le C \exp(-2\,\alpha\, r). \tag{3.32}$$

■

Proof. Part II. For the case of p_o, we may note, under the assumption that $|f(\bar r)| \le C\, \exp(-2\,\alpha' r)$, with $\alpha' > \alpha$, by Lemma 3.4.4 and the identity $d\bar\rho = d\Omega(\bar\rho)\, \rho^2\, d\rho$, that

$$\begin{aligned} |p_o(\bar r)| &= \left| \int_{B_o} \frac{f(\bar r)}{4\,\pi\, |\bar\rho - \bar r|}\, d\bar\rho \right| \\ &\le \int_{|\Omega|=1} \frac{d\Omega}{4\,\pi\, |\bar\rho - \bar r|} \int_r^\infty C\, \exp(-2\,\alpha'\,\rho)\, \rho^2\, d\rho \\ &\le \int_r^\infty C\, \exp(-2\,\alpha'\,\rho)\, \rho\, d\rho, \end{aligned} \tag{3.33}$$

that $|p_o(\bar{r})| \leq C' \exp(-2\,\alpha\, r)$ for some positive constant C'.

■

We have thus established both the analyticity and the asymptotic equality required of Theorem 3.4.1. This completes the proof of this theorem.

■

3.5 The Operation $\mathbf{K}_{x^\ell}$ w

We define the operation $\mathbf{K}_{x^\ell}$ **w** by

$$(\mathbf{K}_{x^\ell}\, w)(\bar{r}, t) = \int_0^t \int_{\mathbb{R}^3} \mathscr{K}_{x^\ell}(\bar{r} - \bar{\rho}, t - t')\, w(\bar{\rho}, t')\, d\bar{\rho}\, dt' \tag{3.34}$$

where $\mathscr{K}$ is defined in (1.20), and the subscripts again indicate differentiation. We shall prove

Theorem 3.5.1. *If $p \in \mathbf{A}^3_{2\,\alpha,d,T}$ and if $T < \infty$, then $\mathbf{K}_{x^\ell}\, p \in \mathbf{A}^3_{2\,\alpha,d,T}$. Moreover,*

$$\|\mathbf{K}_{x^\ell}\, p\,\|_T \leq 2 \left(\frac{T}{\varepsilon\, \pi}\right)^{1/2} \|\Psi|_T. \tag{3.35}$$

Proof. It follows from Lemma 1.5.1 that $\mathscr{K}_{x^\ell}\, p = \mathbf{G}_{x^\ell}\, \mathbf{G}_0\, p$. Theorem 3.5.1 thus follows from Lemma 1.5.1 and Theorem 3.4.1.

■

3.6 Bilinear Forms for Nu

We next examine some bilinear operations that enable more general and also more convenient ways of expressing the operation $\mathbf{N}\,\mathbf{u}$ in Eq. (1.23). To this end, we use *two vectors*, $\mathbf{u} = (u^1,\ u^2,\ u^3)$ and $\mathbf{w} = (w^1,\ w^2,\ w^3)$ to define the following operations:

$$\mathbf{n}(\mathbf{u}, \mathbf{w}) = \mathbf{p}(\mathbf{u}, \mathbf{w}) + \mathbf{q}(\mathbf{u}, \mathbf{w}), \tag{3.36}$$

with

$$\begin{aligned}&\mathbf{p}(\mathbf{u},\mathbf{w})(\bar{r},t)\\&=\int_0^t\int_{\mathbb{R}^3}\left(\left(\nabla'\mathscr{G}(\bar{r}-\bar{r}',t-t')\right)\cdot\mathbf{u}(\bar{r}',t')\right)\mathbf{w}(\bar{r}',t')\,d\bar{r}'\,dt'\\&\mathbf{q}(\mathbf{u},\mathbf{w})(\bar{r},t)\\&=\int_0^t\int_{\mathbb{R}^3}\left(\left(\nabla'\mathscr{K}(\bar{r}-\bar{r}',t-t')\right)f(\mathbf{u},\mathbf{w})(\bar{r}',t')\right)d\bar{r}'\,dt',\end{aligned}\tag{3.37}$$

and with

$$f(\mathbf{u},\mathbf{w})(\bar{r},t)=\sum_{j=1}^{3}\sum_{k=1}^{3}D^{j,k}(u^j\,w^k)(\bar{r},t).\tag{3.38}$$

We also make the following assumptions about $\mathbf{u}$ and $\mathbf{w}$:

Assumption 3.6.1. *We assume that the components of each of the vectors* $\mathbf{u}$ *and* $\mathbf{w}$ *belong to* $\mathbf{A}^3_{\alpha,d,T}$.

3.6.1 The Operation $\mathbf{p}(\mathbf{u},\mathbf{w})$

We prove

Lemma 3.6.1. *If the components of* $\mathbf{u}$ *and* $\mathbf{w}$ *belong to* $\mathbf{A}^3_{\alpha,d,T}$, *then* $\mathbf{p}(\mathbf{u},\mathbf{w})\in\mathbf{A}^3_{2\alpha,d,T}$. *We then furthermore have*

$$|||\mathbf{p}\,(\mathbf{u},\mathbf{w})|||\le 2\left(\frac{T}{\pi\,\varepsilon}\right)^{1/2}|||\mathbf{u}|||\,|||\mathbf{w}|||.\tag{3.39}$$

Proof. The operation $\mathbf{p}(\mathbf{u},\mathbf{w})$ is a vector containing all of the nine terms of the operation $\mathscr{G}_{(x^\ell)'}\sum_{j,k=1}^3 u^j\,w^k$. Thus, for simplicity, we examine an arbitrary one of these, namely, the term

$$p_\ell\,(u^j,w^k)(\bar{r},t)=\int_0^t\int_{\mathbb{R}^3}\mathscr{G}_{(x^\ell)'}(\bar{r}-\bar{r}',t-t')\,(u^j\,w^k)(\bar{r}',t')\,d\bar{r}'\,dt'.\tag{3.40}$$

Since both u^j and w^k belong to $\mathbf{A}^3_{\alpha,d,T}$, the product $u^j\,w^k$ is contained in $\mathbf{A}^3_{2\alpha,d,T}$. Hence we can apply Theorem 3.5.1 to (3.40) to conclude that the term of (3.40) belongs to $\mathbf{A}^3_{2\alpha,d,T}$, to furthermore conclude that $\mathbf{p}(\mathbf{u},\mathbf{w})\in\mathbf{A}^3_{2\alpha,d}$, and also that

$$\|p_\ell\,(u^j,w^k)\|_T\le 2\left(\frac{T}{\pi\,\varepsilon}\right)^{1/2}\|u^j\,w^k\|_T\le\|u^j\|_T\,\|w^k\|_T.\tag{3.41}$$

The identity

$$\sum_{j=1}^{3}\sum_{k=1}^{3} N(u^j)\,N(w^k) = \sum_{j=1}^{3} N(u^j)\sum_{k=1}^{3} N(w^k) = |||\mathbf{u}|||\;|||\mathbf{w}|||, \tag{3.42}$$

now enables us to deduce that

$$|||\mathbf{p}(\mathbf{u},\mathbf{w})||| \le 2\left(\frac{T}{\pi\,\varepsilon}\right)^{1/2} |||\mathbf{u}|||\;|||\mathbf{w}|||. \tag{3.43}$$

■

3.6.2 *The Operation* $\mathbf{q}(\mathbf{u},\mathbf{w})$

We first consider just one of the components of $\mathbf{q}(\mathbf{u},\mathbf{w})$ defined in (3.37), namely,

$$\mathbf{q}_\ell(\mathbf{u},\mathbf{w})(\bar{r},t) = \int_0^t \int_{\mathbb{R}^3} \mathscr{K}_{(x^\ell)'}(\bar{r}-\bar{r}',t-t')\,f(\mathbf{u},\mathbf{w})(\bar{r}',t')\,d\bar{r}'\,dt'. \tag{3.44}$$

Lemma 3.6.2. *If each component of* $\mathbf{u}$ *and* $\mathbf{w}$ *belong to* $\mathbf{A}^3_{\alpha,d,T}$*, then* $\mathbf{q}(\mathbf{u},\mathbf{w}) \in \mathbf{A}^3_{2\alpha,d,T}$*. We then furthermore have*

$$|||\mathbf{q}\,(\mathbf{u},\mathbf{w})|||_T \le 2\left(\frac{T}{\pi\,\varepsilon}\right)^{1/2} |||\mathbf{u}|||_T\;|||\mathbf{w}|||_T. \tag{3.45}$$

Proof. The proof of this lemma follows from Lemma 3.6.1 and Theorem 3.4.1.

■

Upon combining the results of Lemmas 3.6.1 and 3.6.2, we get

Theorem 3.6.2. *If each component of* $\mathbf{u}$ *and* $\mathbf{w}$ *belong to* $\mathbf{A}^3_{\alpha,d,T}$*, then* $\mathbf{n}(\mathbf{u},\mathbf{w}) \in \mathbf{A}^3_{2\alpha,d,T}$*. We then furthermore have*

$$|||\mathbf{n}\,(\mathbf{u},\mathbf{w})|||_T \le 4\left(\frac{T}{\pi\,\varepsilon}\right)^{1/2} |||\mathbf{u}|||_T\;|||\mathbf{w}|||_T. \tag{3.46}$$

Proof. That $\mathbf{Nu} \in \mathscr{B}$ is a consequence of our assumption that $\mathbf{u}(\cdot,t)$ is divergence-free, the determination of g given in (1.14) such that $\mathbf{Nu}$ is then divergence-free.

Theorem 3.6.2 therefore follows from Lemmas 3.6.1 and 3.6.2, since $\mathbf{N}\,\mathbf{u} = \mathbf{n}(\mathbf{u},\mathbf{u})$, while the inequality (3.46) then follows from (3.45).

This completes the proof of Theorem 3.6.2.

■

Let us end this section with a corollary using Eq. (3.15). Upon integrating both sides of this equation over $\mathbb{R}^3$, interchanging the order of integration, and recalling that $\int_{\mathbb{R}^3} z(\bar{\rho})\, d\bar{\rho} = 0$, we deduce the following:

Corollary 3.6.1. *Let* $\mathbf{u} \in \mathscr{B}_T$*, let* g *be as given in (1.14), and let the pressure* $p = G_0\, g$ *be defined as in (1.13). Then*

$$\int_{\mathbb{R}^3} p(\bar{r}, t)\, d\bar{r} = 0. \tag{3.47}$$

Reference

1. http://mathworld.wolfram.com/KummersQuadraticTransformation.html

Chapter 4
Proof of Convergence of Iteration (1.25)

Abstract We prove that if the initial vector $\mathbf{u}^0$ belongs to the Banach space $\mathscr{B}$ defined in Definition 3.1.1, then the iterative sequence of functions defined as in (1.25) converges to a unique solution for all T sufficiently small.

4.1 Contraction Mapping Principle

We consider the definition of a sequence of vector functions $\{U_n\}_{n=0}^{\infty}$ defined as in (1.25). Let the Banach space $\mathscr{B}$ be defined as in Definition 3.1.1, let $\mathbf{u}^0 \in \mathscr{B}$, $|||\{\cdot\}|||_T$ be defined as in (3.1), let $\mathbf{v}$ be defined as in (1.12), and corresponding to a positive number b, let $\mathbf{S}(\mathbf{v}, b)$ denote the family of all vector functions $\mathbf{u}$ such that $|||\mathbf{u} - \mathbf{v}|||_T \leq b$.

The *Contraction Mapping Principle* applied to our operator equation (1.11) takes the following form.

Theorem 4.1.1. *Given the operator equation (1.11), let* $\mathbf{u}^0 \in \mathscr{B}$*, and let the sequence* $\{U_n\}_{n=0}^{\infty}$ *be defined as in (1.25). Let* $\delta \in (0, 1)$ *denote positive number, and set* $b = |||\mathbf{N}(\mathbf{v})|||_T/(1-\delta)$*. Assume that for all* $\mathbf{u}$ *and* $\mathbf{w}$ *in* $\mathbf{S}(\mathbf{v}, b)$*, we have*

$$|||\mathbf{N}(\mathbf{u}) - \mathbf{N}(\mathbf{w})|||_T \leq \delta |||\mathbf{u} - \mathbf{w}|||_T. \tag{4.1}$$

Then $U_n \in \mathbf{S}(\mathbf{v}, b)$ *for all* $n \geq 0$*,* $U_n \to \mathbf{u}^*$ *as* $n \to \infty$*, where* $\mathbf{u}^*$ *is the unique solution in* $\mathbf{S}(\mathbf{v}, b)$ *of Eq. (1.11). Moreover,*

$$\begin{aligned} |||U_n - \mathbf{u}^*|||_T &\leq \frac{\delta}{1-\delta} |||U_{n-1} - U_n|||_T \\ &\leq \frac{\delta^n}{1-\delta} |||\mathbf{N}(\mathbf{v}) - \mathbf{v}|||_T. \end{aligned} \tag{4.2}$$

Proof. We omit the proof, which is obtainable from many sources—see, e.g., Sect. 14.1 of [1].

■

© Springer International Publishing AG 2016

F. Stenger et al., *Navier–Stokes Equations on* $\mathbb{R}^3 \times [0, T]$,
DOI 10.1007/978-3-319-27526-0_4

Let us now determine a number $\delta \in (0,1)$ such that the conditions of Theorem 4.1.1 are satisfied. We can achieve this requirement based on the results of the previous section.

Let $\mathbf{n}(\mathbf{u},\mathbf{w})$ be defined in (3.36), and in this notation, let us replace $\mathbf{u}$ by $\mathbf{u}+\mathbf{w}$, and $\mathbf{w}$ by $\mathbf{u}-\mathbf{w}$. Theorem 3.6.2 then immediately yields:

Lemma 4.1.1. *Let $T>0$, and let each component of $\mathbf{u}(\cdot,t)$ and $\mathbf{w}(\cdot,t)$ belong to the Banach space $\mathscr{B}_T$, as defined in Definition 3.1.1. Then*

$$|||\mathbf{N}(\mathbf{u})-\mathbf{N}(\mathbf{w})|||_T \le c\,T^{1/2}\,|||\mathbf{N}(\mathbf{u}+\mathbf{w})|||_T\,|||\mathbf{u}-\mathbf{w}|||_T, \tag{4.3}$$

where

$$c = 4/(\varepsilon\,\pi)^{1/2}. \tag{4.4}$$

Proof. Upon replacing $\mathbf{u}$ with $\mathbf{u}+\mathbf{w}$ and $\mathbf{w}$ with $\mathbf{u}-\mathbf{w}$ in (3.47), we get (4.3).

■

Next, letting $\mathbf{v}$ be defined as in (1.12), and taking $\mathbf{u}=\mathbf{v}$ in Theorem 3.6.2, we get

Lemma 4.1.2. *Let $\mathbf{u}^0$ belong to $\mathbf{A}^3_{\alpha,d}$ and be divergence-free, let $\mathbf{v}$ and $\mathbf{N}(\mathbf{u})$ be defined in (1.12), and let c be defined in (4.4). Then*

$$|||\mathbf{N}(\mathbf{v})|||_T \le c\,T^{1/2}\,|||\mathbf{v}|||_T^2. \tag{4.5}$$

Next, by repeating the steps of the proofs that lead to the statement of Theorem 3.6.2, we arrive at

Theorem 4.1.2. *If the conditions of Theorem 3.6.2 are satisfied, if $0<\delta\le 2^{-1/2}$, and if*

$$0 < T \le \frac{\delta^2}{32\,|||\mathbf{v}|||_T^4}\,\frac{1-\delta}{1+(1-\delta)^{1/2}}, \tag{4.6}$$

then sequence $\{U_n\}_{n=0}^{\infty}$ defined in (1.25) will converge to the unique solution of Eq. (1.23) as stated in Theorem 4.1.1.

Proof. It follows from Theorems 4.1.1 and 3.6.2 that δ is the maximum value of $c\,T^{1/2}(|||\mathbf{u}+\mathbf{w}|||_T)$, with $\mathbf{u}$ and $\mathbf{w}$ in $\mathbf{S}(\mathbf{v},b)$, and with $b = c\,T^{1/2}\,|||\mathbf{v}|||_T^2/(1-\delta)$. Thus, replacing each of $|||\mathbf{u}|||$ and $|||\mathbf{w}|||$ with $|||\mathbf{v}|||_T+b = |||\mathbf{v}|||_T + c\,T^{1/2}\,|||\mathbf{v}|||_T^2/(1-\delta)$, equating the result to δ, and then solving the resulting quadratic for δ, we get

$$4\,T^{1/2}\,|||\mathbf{v}|||_T^2 = \frac{(1-\delta)^{1/2}}{2}\left(\pm(1+\delta)^{1/2}-(1-\delta)^{1/2}\right). \tag{4.7}$$

Only the (+) sign applies, since the right-hand side of (4.7) must remain positive. we must have $T^{1/2} > 0$. Also, the maximum value of the right-hand side of (4.7) ($\approx$0.2666) occurs for $\delta = 1/\sqrt{2}$.

The statement of Theorem 4.1.2 follows.

■

Reference

1. J. Todd, *Survey of Numerical Analysis* (McGraw-Hill, New York, 1962)

Chapter 5
Numerical Methods for Solving N–S Equations

Abstract In this chapter we describe several methods of approximating solutions to the Navier–Stokes (N–S) integral equations (1.23). We first introduce several possible bases, including polynomial, Sinc, and Fourier polynomials. These bases, covered in Sects. 5.1 and 5.2, enable effective methods for solving N–S equations in the spaces introduced in Chap. 2.These described bases enable several possible ways of carrying out the solution to (1.11) iteratively, and we list some of these in Sects. 5.1 and 5.2.

5.1 Approximating Bases

Let $(w_{-M}, w_{-M+1}, \ldots, w_N)$ be a set of basis functions for interpolating a function f defined on Γ, i.e., an open, finite, semi-infinite, or infinite interval (a, b) on $\mathbb{R}$, or even an analytic arc in the complex plane, i.e.,

$$f(x) \approx \sum_{k=-M}^{N} w_k(x) f(x_k), \quad x \in \Gamma. \tag{5.1}$$

Here, the x_k are distinct points, which we assume to appear on Γ in the order $x_{-M}, x_{-M+1}, \ldots, x_N$. The w_k are basis functions, which can be polynomials, rational functions, Sinc functions, trigonometric functions, etc.

5.1.1 Bases for Solving N–S Equations

Various methods of approximation may be used to approximate solutions to Navier–Stokes Equations.

5.1 (i) Using *polynomials* p_m with zeros at
$x_{-M}, x_{-M+1}, \ldots, x_N$, where $m = M + N + 1$. We define the w_k as follows:

© Springer International Publishing AG 2016

F. Stenger et al., *Navier–Stokes Equations on* $\mathbb{R}^3 \times [0, T]$,
DOI 10.1007/978-3-319-27526-0_5

$$w_k(x) = \frac{p_m(x)}{(x - x_k)\, p'_m(x_k)} = \prod_{\ell=-M, \ell\neq k}^{N} \frac{x - x_\ell}{x_k - x_\ell}. \tag{5.2}$$

5.1 (ii) For *Sinc interpolation* on Γ, we use a one-to-one transformation φ of $\Gamma \to \mathbb{R}$ to get an approximation of the form (5.1), with

$$w_k(x) = \frac{\sin\left\{\frac{\pi}{h}(\varphi(x) - k\,h)\right\}}{\frac{\pi}{h}(\varphi(x) - k\,h)}. \tag{5.3}$$

Such methods and the error of their approximations are discussed at length in [8, 11], and we will not go into further details here.
5.1 (iii) Similarly, *trigonometric wavelet formulas* may be used. Details for these can be found in Sect. 1.4 of [11]. Such methods are best suited for use with Fourier transforms and FFT, and for interpolating periodic functions.
5.1 (iv) To date, we have not yet investigated the use of splines; we leave open this possibility for future research.

For all of these cases, we define a row vector $\mathbf{w}$ of basis functions and an operator $\mathbf{V}$ by

$$\mathbf{w} = (w_{-M}, \ldots, w_N)\,, \quad \mathbf{V}\, g = (g(x_{-M}), \ldots, g(x_N))^T\,; \tag{5.4}$$

these enable us to write (5.1) in operator form,

$$f \approx \mathbf{w}\,\mathbf{V} f. \tag{5.5}$$

5.2 Approximation Operations on Bases

We shall use several operations on bases.

5.2.1 DFT Approximation

We start with a Sinc approximation of a function f defined on $\mathbb{R}$, which we approximate via a truncated Sinc expansion in the form

$$f(x) \approx v(x) = \sum_{k=-N}^{N} v(k\,h_x)\, S(k, h)(x). \tag{5.6}$$

where, for convenience, setting $f_j = f(j\,h_x)$, for $j = -N,\ \ldots,\ N$, we define $\mathbf{v}$ and its Fourier transform $\hat{\mathbf{v}}$ by

$$\begin{aligned} \mathbf{v} &= ((1/2)f_{-N}, f_{-N+1},\ \ldots,\ f_{N-1},\ (1/2)f_N)^T\,, \\ \hat{\mathbf{v}} &= (\hat{v}_{-N},\ \ldots,\ \hat{v}_N)^T\,. \end{aligned} \tag{5.7}$$

If $f \in \mathbf{A}^1_{\alpha,d}$ and we take $h = h_x = (\pi\, d/(\alpha\, N))^{1/2}$ then the $\mathbf{L}^p(\mathbb{R})$ difference between f and v is of the order of $N^{1/2}\,\exp\left(-(\pi\, d\,\alpha\, N)^{1/2}\right)$ for $p = 2$ and also for $p = \infty$ (see [8], Sect. 3.3). This error bound extends readily to more than one dimension—see ([8], Theorem 6.6.2). Next, taking the 1-d Fourier transform of each side of (5.6) we get the DFT approximation of the Fourier transform $\hat{f}$ of f, i.e., with $y \in \mathbb{R}$,

$$\begin{aligned} \hat{f}(y) &= \int_{\mathbb{R}} e^{i\,x\,y} f(x)\,dx \\ &= \hat{f}(y) \approx \hat{v}(y) = \begin{cases} \displaystyle\sum_{k=-N}^{N} v(k\,h_x)\,e^{i\,k\,h\,y} & \text{if } |y| < \frac{\pi}{h} \\ 0 \text{ if } |y| > \frac{\pi}{h}. \end{cases} \end{aligned} \tag{5.8}$$

Note that in the Fourier transform space of variable y, there are N equi-spaced intervals on $(0, \pi/h_x)$, and hence each of these has size $h_y = \pi/(N\,h_x)$. The difference on $\mathbb{R}$ between $\hat{f}$ and $\hat{v}$ is again of the order of $N^{1/2}\,\exp\left(-(\pi\, d\,\alpha\, N)^{1/2}\right)$ for $p = 2$ and also for $p = \infty$ (see [8], Sect. 3.3).

Equations (5.6)–(5.8) enable us to define two DFT matrices, $A = [a_{j,k}]$ and $B = [b_{j,k}]$, with

$$\begin{aligned} a_{j,k} &= h_x\,\exp(i\,j\,k\,\pi/N) \\ b_{j,k} &= \frac{1}{2\,N\,h_x}\,e^{-i\,j\,k\,\pi/N}, \end{aligned} \tag{5.9}$$

Let us illustrate explicitly the use of these matrices.

Letting $\mathbf{v}$ be defined as above, we get

$$\begin{aligned} \hat{\mathbf{v}} &= A\,\mathbf{v} \\ \mathbf{v} &= B\,\hat{\mathbf{v}}. \end{aligned} \tag{5.10}$$

More details of these derivations are given in [11, Sect. 1.4.9].

5.2.2 *Indefinite Integration*

The *indefinite integration operators* are defined on $\Gamma = (a, b)$ by

$$(\mathscr{J}^{+}g)(x) = \int_a^x g(t)dt, \quad (\mathscr{J}^{-}g)(x) = \int_x^b g(t)dt. \tag{5.11}$$

Using the basis function w_k defined in (5.2) or (5.3) above (or even one of the trigonometric wavelet bases of [11], Eq. (1.4.16) or [11], Eq. (1.4.19), we define numbers $W_{jk}^{\pm}$ $(j, k) \in (-M, \ldots, N)$ which are the (j, k)th entries of $m \times m$ ($m = M + N + 1$) matrices $\mathbf{W}^{\pm}$, by

$$\begin{aligned} W_{jk}^{+} &= \int_a^{x_j} w_k(x)\,dx, \quad W_{jk}^{-} = \int_{x_j}^b w_k(x)\,dx, \\ \mathbf{W}^{\pm} &= \left[W_{jk}^{\pm}\right]. \end{aligned} \tag{5.12}$$

It is convenient to also introduce definitions of the corresponding approximating operators $\mathscr{J}_m^{\pm}$, which yield the accurate approximations

$$\mathscr{J}^{\pm} g \approx \mathscr{J}_m^{\pm} g; \tag{5.13}$$

these are defined by

$$\mathscr{J}_m^{\pm} g = \mathbf{w}\,\mathbf{W}^{\pm}\,\mathbf{V}\,g, \tag{5.14}$$

where $\mathbf{w}$ and $\mathbf{V}$ are defined in (5.4) above. These approximations are accurate in the spaces of Chap. II, and the error of these approximations are discussed in Sect. 4.5 of [8] (and in Sect. 1.4 of [11]) for the case of trigonometric wavelet approximation.

Corresponding to the exact expression, $G = \mathscr{J}^{\pm} f$, we obtain, upon applying the definition of $\mathbf{V}$ given in (5.4)–(5.9), that

$$\mathbf{V}\,G \approx \mathbf{V}\left(\mathscr{J}_m\, g\right) = \mathbf{W}^{\pm}\mathbf{V}\,g. \tag{5.15}$$

The eigenvalues of the matrices $W^{\pm}$ have some very pleasant and useful properties. These properties stated in the theorem which follows were conjectured for the case of Sinc approximation by Stenger over 25 years ago; Stenger then offered \$300 for proof or disproof of this conjecture. A proof was recently found by Han and Xu [5].

Theorem 5.2.1. *The matrices* $\mathbf{W}^{+}$ *and* $\mathbf{W}^{-}$ *have the same eigenvalues* s_j*, but they have different eigenvectors. All of the* $m = M + N + 1$ *eigenvalues of Sinc approximation matrices* $\mathbf{W}^{\pm}$ *lie on the open right half plane.*

5.2.3 Differentiation

The derivative of the Sinc series for a function f on $\mathbb{R}$ yields an accurate approximation of f', at least in the case when f belongs to a Sinc space $\mathbf{A}^1_{\alpha,d}$. However, the derivative of a Sinc approximation is not accurate in the neighborhood of a finite end-point of an interval of approximation, although in this case, the derivative of polynomial that interpolates at Sinc points does yield an accurate approximation of the derivative of f [10]. Furthermore, since the eigenvalues of the indefinite integration matrices lie on the open right half plane, the inverses of these matrices for both polynomial and Sinc approximation can be used to get accurate approximations to the derivative, i.e., if $G = \mathscr{J}^{\pm} g$, we get the accurate approximation at interpolation points,

$$\mathbf{V}\, g \approx \pm\, (\mathbf{W}^{\pm})^{-1} \mathbf{V}\, G. \tag{5.16}$$

5.2.4 Indefinite Convolution

See, ([8], Sect. 4.6), or ([11], Sect. 1.5.9). Our model integrals over an interval $(a, b) \subseteq \mathbb{R}$ take the form

$$\begin{aligned} p(x) &= \int_a^x f(x-t)\, g(t)\, dt, \\ q(x) &= \int_x^b f(t-x)\, g(t)\, dt. \end{aligned} \tag{5.17}$$

Taken as integral equations, these two are the most often solved equations in the world, e.g., g, given p (or q) and f (think, e.g., of using the *p.a.d.* of a smart phone, pressing a typewriter key, or pressing a button in an elevator). However, whereas such modern devices use chips based on ODE with constant coefficients, our methods for evaluating (or solving) (5.17) have advantages, in that they are not restricted in this way.

The *Laplace transform* $f(s)$ and—our preferred version—the "Laplace transform" $F(s)$ are given as integrals over $(0, c)$ with $c \geq b - a$ by

$$\begin{aligned} f(s) &= \int_0^c e^{-st} f(t)\, dt \\ F(s) &= \int_0^c e^{-t/s} f(t)\, dt, \end{aligned} \tag{5.18}$$

where *we assume that these exists for all* $\Re s > 0$. We may note also that the mapping $s \to 1/s$ is a one-to-one transformation of the right half of the complex plane to itself.

The following has been proved [8–10]:

Theorem 5.2.2. *Let p, q, g, F, and* $\mathscr{J}^{\pm}$ *be defined as above. Then*

$$p = F\left(\mathscr{J}^{+}\right) g, \quad q = F\left(\mathscr{J}^{-}\right) g. \tag{5.19}$$

Proof. We omit the proof of this theorem, which can be found in the above cited references.

■

Equation (5.19) appears esoteric. They are, nevertheless, extremely useful. To illustrate, let $\mathbf{w}$, $\mathbf{V}$, and $\mathbf{W}^{+}$ be defined as above, and let, e.g., $\mathbf{W}^{+} = X S X^{-1}$, where $S = \text{diag}\,(s_{-M}, \ldots, s_N)$ is a diagonal matrix and where X is the corresponding matrix of eigenvectors. Since we have the accurate approximations $\mathscr{J}^{\pm} \approx \mathscr{J}_m^{\pm}$ for the case when these operators are applied to functions in $\mathbf{A}^1_{\alpha,d}$, we have the similarly accurate approximations $\mathscr{J}_m^{\pm} g \approx \mathscr{J}^{\pm} g$. More explicitly, e.g., for the case of $\mathscr{J}^{+} g$,

$$\begin{aligned} p &= F\left(\mathscr{J}^{+}\right) g \\ &\approx F\left(\mathscr{J}_m^{+}\right) g \\ &\approx \mathbf{w}\, F(\mathbf{W}^{+})\, \mathbf{V}\, g \\ &= \mathbf{w}\, X\, F(S)\, X^{-1}\, \mathbf{V}\, g. \end{aligned} \tag{5.20}$$

That is, in matrix-vector form, we get the accurate approximation

$$\mathbf{V}\, p \approx X\, F(S)\, X^{-1}\, \mathbf{V}\, g, \tag{5.21}$$

at the points at which we interpolated g. Similarly for the case of q.

Remark 5.2.1.

5.2.1(i) **The diagonal matrix** $F(S)$**.** The numbers $F(s_j)$ denote the elements of the diagonal matrix S. If F is not explicitly known, then the values $F(s_j)$ can be approximated to arbitrary accuracy via use of Sinc quadrature applied to the "Laplace transform" integral (see [11]).

5.2.1(ii) **The matrices** $W^{\pm}$**.** The operators $\mathscr{J}^{\pm}$ and the above matrices $\mathbf{W}^{\pm}$ are translation invariant, i.e., they are the same for $(a + c, b + c)$ for any $c \in \mathbb{R}$. The bases $\mathbf{w}$ will, of course, have to be modified depending on the interval, i.e., on $(a + c, b + c)$ the basis is $\mathbf{v}$, with $v_j(x) = w_j(x - c)$.

5.2.1(iii) **Positivity of the real parts of the eigenvalues** s_j**.** Recall, we assumed that $\Re s > 0$ for the existence of the Laplace transform (5.13) above. We thus require that all eigenvalues of $\mathbf{W}^{\pm}$ have positive real parts, in order for the approximation in (5.20) to be well defined. The Conjecture A referred to above alludes to this for orthogonal polynomial methods, while the recent proof [5] shows that this is in fact the case for Sinc approximation.

The above convolution approximation formulas have important ramifications in the following cases:

5.2.1(iv) **Abel integrals**, or **Abel integral equations.** These are no longer a problem. See the example in Sect. 3.6 of [11] and [1–3].

5.2.1(v) **Laplace transform inversion.** The above convolution procedure yields a new formula for Laplace transform inversion (Sect. 1.5.10 of [11]). If $g(s) = \int_0^c e^{-st} f(t)dt$, for some $c > 0$, and if (e.g.) $\mathscr{J}^+$ is the indefinite integral operator over $(0, d)$, for some $d \leq c$, then we get the following identity ([8], Sect. 4.6), or ([11], Sect. 1.5.9) for f over $(0, d)$:

$$f = \left(\mathscr{J}^+\right)^{-1} g\left(\left(\mathscr{J}^+\right)^{-1}\right) 1, \tag{5.22}$$

where "1" is the identically 1 function on $(0, d)$.

If $\mathbf{W}^+$ is defined as in (5.11) above, then we have the following easy to obtain approximation of f at the interpolation points $\{x_j\}_{j=-M}^N$

$$\mathbf{V}f \approx \left(\mathbf{W}^+\right)^{-1} g\left(\left(\mathbf{W}^+\right)^{-1}\right) \mathbf{1}, \tag{5.23}$$

where $\mathbf{1}$ is a vector of $m = M + N + 1$ ones. This formula enables us to compute f to arbitrary accuracy on $(0, d)$, provided that f is analytic on this interval. It is a stable formula, and except for "table lookup," it is considerably easier to apply than any other inversion formula known to us. Note, (5.15) enables us to write (5.17) in the more computable form

$$\mathbf{V}f \approx X\left(S^{-1} F(S)\right)\left(X^{-1}\mathbf{1}\right). \tag{5.24}$$

That is, if a user is happy with a certain approximation that is determined by the (usually relatively small) matrix size $m = M+N+1$, he/she can store the matrix X as well as the vectors $u = (s_{-M}, \ldots, s_N)^T$ and $v = X^{-1}\mathbf{1}$. Once he obtains $f(s)$, he can form the vector

$$v = \left(s_{-M}^{-1} f\left(s_{-M}^{-1}\right), \ldots, s_N^{-1} f\left(s_N^{-1}\right)\right)^T,$$

form the Hadamard product $w = u * v$ of the two vectors u and v, and then form the product $\mathbf{f} = Xw$ for which the vector $\mathbf{f} = (f_{-M}, \ldots, f_N)^T$ yields the approximations $f_j \approx f(x_j)$, where x_j is defined as in Sect. 5.1 above. The amount of work is thus not much different from performing FFT. Furthermore our family of approximation methods enables accurate approximations even though f may have singularities at the end-points a or b (or both), and in our methods enable accurate approximation with matrices of small order m even though f may approach zero slowly.

5.2.1(vii) **Hilbert transforms.** The identities (5.18) enable a new formula expression for Hilbert transforms (see [8] or Sect. 1.5.12 of [11]) on (a, b). It is the following:

$$(\mathscr{H}\, g)(x) = P.V. \int_a^b \frac{g(t)}{x-t}\, dt$$
$$= \left(\log\left(\mathscr{J}^+\right) - \log\left(\mathscr{J}^-\right)\right) g \tag{5.25}$$

$$\mathbf{V}\,\mathscr{H}\, g \approx \left(\log\left(\mathbf{W}^+\right) - \log\left(\mathbf{W}^-\right)\right) \mathbf{V}\, g.$$

This identity, together with the easy to compute formula for the index ([8], Theorem 1.3.2), enables easy to apply numerical procedures for solving any of the singular integral equation (including as Wiener–Hopf; see below) and Riemann problems toward which Nöther [7] made important contributions [4, 6].

5.2.1(viii) **Wiener–Hopf Problems.** The classical Wiener–Hopf integral equation takes the form

$$f(x) - \int_0^\infty k(x-t) f(t)\, dt = g(x), \quad x \in (0, \infty). \tag{5.26}$$

The problem is to solve this equation for f, given k and g.

During the last few years of the 1960's decade, pure and applied mathematicians of nearly all of the "top 10" mathematics departments in the US worked on problems related to the Wiener–Hopf method. During that time, the pure mathematicians found a number of excellent novel results on this subject. However, the problem was not solved in the applied world during that time, since it is almost never possible to find the requisite explicit factorization of $1-K$, with K denoting the Fourier transform of k. The above indefinite convolution procedure enables us to circumvent such a factorization, by proceeding as follows.

Write Eq. (5.26) in the form

$$f(x) - \int_0^x k(x-t) f(t)\, dt - \int_x^\infty k(x-t) f(t)\, dt = g(x), \quad x > 0. \tag{5.27}$$

At this point we can apply indefinite convolution to each of these integrals to get the linear system of algebraic equations

$$\mathbf{V} f - \left(F^+(\mathbf{W}^+) + F^-(\mathbf{W}^-)\right) \mathbf{V} f \approx \mathbf{V}\, g, \tag{5.28}$$

where

$$F^{\pm}(s) = \int_0^\infty e^{-t/s}\, k(\pm t)\, dt. \tag{5.29}$$

The solution (5.27) enables arbitrary accuracy, even (via use of singular value decomposition) when the IE (5.26) does not have a unique solution. Indeed, this method of solution extends to solving integral equation formulations of partial differential equation problems.

5.2.1(ix) **Novel methods solving PDE.** The above indefinite convolution procedure extends to more than one dimension, in which case it enables approximate solutions of all PDE of the form

$$\mathbf{L}\,u = g, \tag{5.30}$$

where, in 1, 2, or 3 space dimensions (and 4 dimensions including time, for the case of heat and wave problems), $\mathbf{L}\,u$ can take any of the forms

$$\mathbf{L}\,u = \begin{cases} -\,\Delta\,u, \\ \dfrac{1}{\varepsilon}\,\dfrac{\partial u}{\partial t} - \Delta\,u, \ \text{or} \\ \dfrac{1}{c^2}\,\dfrac{\partial^2 u}{(\partial t)^2} - \Delta\,u. \end{cases} \tag{5.31}$$

$$= g(\bar{r}, t, u, \nabla u).$$

All such PDE can be reduced to the solution of integral equation problems using Green's functions. By the above convolution procedure, we can treat the time variable the same as a space variable. Furthermore, for each (space or space plus time) dimension, each of the above PDE is solved via use of one-dimensional matrix multiplication algorithms that differ only in the different forms of the "Laplace transforms" of the Green's functions.

5.2.1(x) *Indefinite Convolution Methods for Navier–Stokes Problems.* Let us next mention some methods which we shall apply in this monograph to solve the problem (1.23).

5.3 Approximating the N–S Equations

There are many ways of approximating the operators in Eqs. (1.23)–(1.24), some of these being better than others. Since we have shown that the solution belongs to $\mathbf{A}^3_{\alpha,d}$ with respect to the spacial variable $\bar{r}$, indefinite convolution combined with Sinc approximation works in all cases. DFT approximation on $\mathbb{R}^3$ meshes with Sinc approximation; it is suitable in all but one case, namely, that of the Fourier transform $\widehat{\mathbf{G}}_0\,w$, due to the presence of the denominator Λ^2 in this expression. We thus approximate the pressure using indefinite convolution combined with Sinc approximation. We are also restricted to using Sinc approximation in the variable t on the finite interval $[0, T]$, since the solution of Eq. (1.11) has a singularity at $t = 0$.

Let us determine our notation for the interpolation points, for approximation over $\mathbb{R}^3 \times [0, T]$. For approximation as a function of $\bar{r} = (x, y, z)$, we shall interpolate the solution with spacing $h_{\bar{r}} = h$ on each interval $[-X/2 \le x^j \le X/2]$, so that

$h = X/(2N)$. The Fourier transforms of Sinc approximation vanishes outside of each interval $[-\pi/h, \pi/h] = [-Y/2, Y/2]$, so that we here have equi-space approximation with spacing $h_{\bar{\Lambda}} = h' = \pi/(N h)$, i.e., $Y = 2N/h$. The points of interpolation in the t variable are just t_μ, where

$$t_\mu = \frac{T \exp(\mu\, h_T)}{1 + \exp(\mu\, h_T)}, \quad \mu = -M, \ \ldots, M. \tag{5.32}$$

We shall also use the τ and $\tau_{\mu'}$ notations in the one-dimensional convolution approximation of the convolutions with respect to t.

We also introduce a more convenient notation for interpolation in $\mathbb{R}^3$ and the 3–d Fourier transform space, as follows,

$$\begin{aligned} \bar{r} &= \bar{r}_{j,k,\ell} = h\,(j,k,\ell), \\ \bar{\Lambda} &= \bar{\Lambda}_{j,k,\ell} = h'\,(j,k,\ell), \\ \Lambda &= \Lambda_{j,k,\ell} = h'\left(j^2 + k^2 + \ell^2\right)^{1/2}, \end{aligned} \tag{5.33}$$

with $h' = \pi/(N h_x)$.

5.3.1 *The Vector* **v**

The Fourier transform of a component v^ν of **v** is given by

$$\hat{v}^\nu(\bar{\Lambda}, t) = \hat{u}^{0,\nu}(\bar{\Lambda}) \exp(-\Lambda^2\, \varepsilon\, t), \tag{5.34}$$

which yields the approximation

$$\hat{V}^\nu = \left[\hat{v}^\nu_{j,k,\ell,\mu}\right] = \left[\hat{u}^{0,\nu}_{j,k,\ell}\, \hat{G}_{j,k,\ell,\mu}\right], \tag{5.35}$$

where

$$\left[\hat{G}_{j,k,\ell,\mu}\right] = \left[\exp(-(\Lambda_{j,k,\ell})^2\, \varepsilon\, t_\mu)\right]. \tag{5.36}$$

5.3.2 *The Approximation of* $\mathbf{P}\,\mathbf{u} + \mathbf{Q}\,\mathbf{u}$

Let us observe, at the outset, that from (1.28) and the definition $\mathscr{K} = \mathscr{G} * \mathscr{G}_0$ of (1.19), we have that the 3-d Fourier transform of the νth equation of (1.26) is given for $\nu = 1, 2, 3$ by

$$\begin{aligned}\widehat{\mathbf{N}^{\nu}\,\mathbf{u}}(\bar{\Lambda}, t) &= \widehat{(\mathbf{P}^{\nu}\,\mathbf{u} + \mathbf{Q}^{\nu}\,\mathbf{u})}(\bar{\Lambda}, t) \\ &= \int_0^t \hat{\mathbf{G}}(\bar{\Lambda}, t - t') \left(\sum_{j=1}^{3} \left(i\,\lambda^j\, \widehat{u^j * u^{\nu}}(\bar{\Lambda}, t') \right) + i\,\lambda^{\nu}\,\hat{p}(\bar{\Lambda}, t') \right) dt',\end{aligned} \tag{5.37}$$

where the "*" denotes a Hadamard product, where p denotes the pressure

$$p = \mathbf{G}_0 f, \tag{5.38}$$

where $f = f(\mathbf{u}, \mathbf{u})$, and with $f(\mathbf{u}, \mathbf{w})$ given in (3.38).

It is evidently convenient to first compute the pressure p, in view of (5.37) and (5.38). We shall carry out this operation using 3-d Sinc indefinite convolution by means of the following result (see [8, 9, 12]), about which we omit the details although we present these details in our algorithm and program in Sect. 5.4 and in Appendix A below.

Let us here state for completeness the 3-d "Laplace transform" of the Green's function $G_0(\bar{r}) = 1/(4\,\pi\, r)$.

Lemma 5.3.1. *Set* $\bar{\rho} = \left(1/\xi^1,\ 1/\xi^2,\ 1/\xi^3\right)$, *and* $\rho = \left(1/(\xi^1)^2 + 1/(\xi^2)^2 + 1/(\xi^3)^2\right)^{1/2}$. *Then*

$$\begin{aligned}G(\bar{\rho}) = G(\xi^1, \xi^2, \xi^3) &= \int_0^{\infty} \int_0^{\infty} \int_0^{\infty} \frac{1}{4\,\pi\, r} \exp(-\bar{r} \cdot \bar{\rho})\, d\bar{r} \\ &= -\frac{1}{\rho^2} \left(\frac{1}{8} - H_0(\xi^1, \xi^2, \xi^3) - H_0(\xi^2, \xi^3, \xi^1) - H_0(\xi^3, \xi^1, \xi^2) \right).\end{aligned} \tag{5.39}$$

Letting $Q(a)$ be defined as in ([11], Theorem 2.4.1), and setting

$$\begin{aligned}\eta^1 &= \sqrt{\frac{1}{(\xi^2)^2} + \frac{1}{(\xi^3)^2}}, \\ \zeta^1 &= \sqrt{\frac{\xi^3 + i\,\xi^2}{\xi^3 - i\,\xi^2}},\end{aligned} \tag{5.40}$$

we have

$$H_0(\xi^1, \xi^2, \xi^3) = +\frac{1}{4\,\pi\,\xi^1\,\eta^1}\left(Q(i\,\zeta^1) - Q(-i\,\zeta^1)\right). \tag{5.41}$$

The use of this identity was already demonstrated to be effective in [12].

5.4 A More Detailed Algorithm

Initializations

1. We first select an error criterion ε, and then we determine space parameters α, d, h, N, and time parameters α_T, d_T and h_T and M to achieve an error of

$$\begin{aligned} \varepsilon &\leq N^{1/2}\,\exp\left(-\left(\frac{\pi\,d}{\alpha\,N}\right)^{1/2}\right) \\ &\approx M^{1/2}\,\exp\left(-\left(\frac{\pi\,d_T}{\alpha_T\,N}\right)^{1/2}\right); \end{aligned} \tag{5.42}$$

2. We then also determine the spacial ($x_j^{\nu} = j\,h$) and time variable (t_k—see (5.32)) Sinc points, the Fourier transform variable Sinc points $y_j^{\nu} = j\,h'$, a DFT matrix of order $m = 2\,N + 1$, and its inverse, an indefinite integration matrices $A = A_T = Z\,S_T\,Z^{-1}$ of order $m_T = 2\,M + 1$ and $B = B_{\bar{r}} = Y\,S_{\bar{r}}\,Y$ as well as the eigenvector matrix Z and its inverse, and also, its corresponding eigenvalues $S_T = \text{diag}\,(\tau_{-M},\ \ldots,\ \tau_M)$ and $S_{\bar{r}} = \text{diag}\,(s_{-N},\ \ldots,\ s_N)$.
3. We then also evaluate the 3-d arrays $U^{0,\nu} = \left[u^{0,\nu}(j\,h, k\,h, \ell\,h)\right]$ for $\nu = 1,\ 2,\ 3$, as well as the Fourier transform arrays $\widehat{U^{0,\nu}}$.
4. We then also need to determine the Fourier transforms of each of the components v^{ν} of $\mathbf{v}$, and evaluate these at the points $\bar{\Lambda}_{j,k,\ell} = h'\,(j, k, \ell)$. Since the Fourier transform of $\mathscr{G}$ is $\exp(-\Lambda^2\,\varepsilon\,t)$, these four-dimensional arrays are given by

$$\widehat{V^{\nu}} = \left[\widehat{V^{\nu}}_{j,k,\ell\mu}\right] = \left[\exp\left(-\Lambda^2_{j,k,\ell}\,\varepsilon\,t_{\mu}\right)\ \widehat{U^{\nu}}_{j,k,\ell}\right];$$

5. Set $\widehat{W^{\nu}} = \widehat{V^{\nu}}$, $\nu = 1, 2, 3$;
6. (****) Set $\widehat{U^{\nu}} = \widehat{W^{\nu}}$, $\nu = 1, 2, 3$;

The First Equation of (1.26)

We perform the following evaluations.

1. Form the Hadamard products $X^1 = U^1 * U^1$, $X^2 = U^1 * U^2$, $X^3 = U^1 * U^3$, $X^4 = U^2 * U^2$, $X^5 = U^2 * U^3$, and $X^6 = U^3 * U^3$;
2. Form the 3-d Fourier transforms $\widehat{X^{\nu}}$ of these;

3. Form the 4-d array

$$\hat{f} = -\left((\lambda^1)^2\,\widehat{X^1} + 2\,\lambda^1\,\lambda^2\,\widehat{X^2} + 2\,\lambda^1\,\lambda^3\,\widehat{X^3}\right)$$

$$-\left((\lambda^2)^2\,\widehat{X^4} + 2\,\lambda^2\,\lambda^3\,\widehat{X^5} + (\lambda^3)^2\,\widehat{X^6}\right);$$

4. Form f from $\hat{f}$ via 3-d spacial inverse Fourier transform. Note, this has to be done with respect to each t_μ to get the complete 4-d array;
5. Form $p = \mathbf{G}_0 f$, using 3-d Sinc convolution. This also needs to be done for each t_μ;
6. Form $\hat{p}$ from p via 3-d spacial Fourier transform, and then form $\hat{y} = i\,\lambda^1\,\hat{p}$;
7. Form $\hat{z} = i\left(\lambda^1\,\widehat{X^1} + \lambda^2\,\widehat{X^2} + \lambda^3\,\widehat{X^3}\right)$;
8. Form $\hat{x} = \hat{y} + \hat{z}$;
9. For $\{j,k,l\}_{-N,-N,-N}^{N,N,N}$ do the following:
   a. Form the vector $\hat{e} = \hat{x}_{j,k,l,\cdot}$ of size $m = 2M+1$ from the array $\hat{x}$;
   b. Form diagonal matrix $D = diag(d_{-M},\ \ldots,\ d_M)$ with

$$d_\mu = \frac{\tau_\mu}{1 + \varepsilon\,\Lambda_{j,k,l}^2\,\tau_\mu};$$

   c. Form the matrix $C = Z\,D\,Z^{-1}$,—see above for the definition of Z;
   d. Replace $\hat{e}$ with $\hat{e} = C\,\hat{e}$;
   e. Form $\hat{N}_{j,k,l,\cdot} = \hat{e}$
10. Form $\widehat{U^1} = \widehat{V^1} + \hat{N}$;
11. Form U^1 from $\widehat{U^1}$ via 3-d spacial inverse Fourier transform;

The Second Equation of (1.26)

We next perform the following evaluations.

1. Form the Hadamard products $X^\nu = U^1 * U^\nu$, for $\nu = 1, 2, 3$. Note: X^4, X^5, and X^6 are the same as for the first equation above;
2. Form the 3-d spacial Fourier transforms $\widehat{X^\nu}$ of these three new 4-d arrays;
3. Form the 4-d array

$$\hat{f} = -\left((\lambda^1)^2\,\widehat{X^1} + 2\,\lambda^1\,\lambda^2\,\widehat{X^2} + 2\,\lambda^1\,\lambda^3\,\widehat{X^3}\right)$$

$$-\left((\lambda^2)^2\,\widehat{X^4} + 2\,\lambda^2\,\lambda^3\,\widehat{X^5} + (\lambda^3)^2\,\widehat{X^6}\right);$$

4. Form f from $\hat{f}$ via 3-d spacial inverse Fourier transform. Note, this has to be done with respect to each t_μ to get the complete 4-d array;
5. Form $p = \mathbf{G}_0 f$, using 3-d Sinc convolution. This needs to be done for each t_μ;
6. Form $\hat{p}$ from p via 3-d spacial Fourier transform, and then form $\hat{y} = i\,\lambda^2\,\hat{p}$;
7. Form $\hat{z} = i\,\left(\lambda^1\,\widehat{X^2} + \lambda^2\,\widehat{X^4} + \lambda^3\,\widehat{X^5}\right)$;
8. Form $\hat{x} = \hat{y} + \hat{z}$;
9. For $\{j,k,l\}_{-N,-N,-N}^{N,N,N}$ do the following:
   a. Form the vector $\hat{e} = \hat{x}_{j,k,l,\cdot}$ of size $m = 2M+1$ from the array $\hat{x}$;
   b. Form diagonal matrix $D = diag(d_{-M},\ \ldots,\ d_M)$ with

$$d_\mu = \frac{\tau_\mu}{1 + \varepsilon\,\Lambda_{j,k,l}^2\,\tau_\mu}.$$

   Note these entries are the same as for the first equation;
   c. Form the matrix $C = Z\,D\,Z^{-1}$. See above for the definition of Z;
   d. Replace $\hat{e}$ with $\hat{e} = C\,\hat{e}$;
   e. Form $\hat{N}_{j,k,l,\cdot} = \hat{e}$
10. Form $\widehat{U^2} = \widehat{V^2} + \hat{N}$;
11. Form U^2 from $\widehat{U^2}$ via 3-d spacial inverse Fourier transforms;

The Third Equation of (1.26)

We next perform the following evaluations.

1. Form the Hadamard products $X^2 = U^1 * U^2, X^4 = U^2 * U^2$, and $X^5 = U^2 * U^3$. The remaining X^j's are the same as for the second equation above;
2. Form the 3-d Fourier transforms $\widehat{X^\nu}$ of these three new 4-d arrays;
3. Form the 4-d array

$$\begin{aligned}\hat{f} = &-\left((\lambda^1)^2\,\widehat{X^1} + 2\,\lambda^1\,\lambda^2\,\widehat{X^2} + 2\,\lambda^1\,\lambda^3\,\widehat{X^3}\right)\\ &-\left((\lambda^2)^2\,\widehat{X^4} + 2\,\lambda^2\,\lambda^3\,\widehat{X^5} + (\lambda^3)^2\,\widehat{X^6}\right);\end{aligned}$$

4. Form f from $\hat{f}$ via 3-d spacial inverse Fourier transform. Note, this has to be done with respect to each t_μ to get the complete 4-d array;
5. Form $p = \mathbf{G}_0 f$, using 3-d Sinc convolution. This needs to be done for each t_μ;
6. Form $\hat{p}$ from p via 3-d spacial Fourier transform, and then form $\hat{y} = i\,\lambda^3\,\hat{p}$;
7. Form $\hat{z} = i\,\left(\lambda^1\,\widehat{X^3} + \lambda^2\,\widehat{X^5} + \lambda^3\,\widehat{X^6}\right)$;
8. Form $\hat{x} = \hat{y} + \hat{z}$;
9. For $\{j,k,l\}_{-N,-N,-N}^{N,N,N}$ do the following:
   a. Form the vector $\hat{e} = \hat{x}_{j,k,l,\cdot}$ of size $m = 2M+1$ from the array $\hat{x}$;
   b. Form diagonal matrix $D = diag(d_{-M},\ \ldots,\ d_M)$ with

$$d_\mu = \frac{\tau_\mu}{1 + \varepsilon\, \Lambda_{j,k,l}^2\, \tau_\mu}.$$

Note these entries are the same as for the first equation;

c. Form the matrix $C = Z D Z^{-1}$—see above for the definition of Z;
d. Replace $\hat{e}$ with $\hat{e} = C\,\hat{e}$;
e. Form $\hat{N}_{j,k,l,\cdot} = \hat{e}$

10. Form $\widehat{U^3} = \widehat{V^3} + \hat{N}$;
11. Form U^3 from $\widehat{U^3}$ via 3-d spacial inverse Fourier transforms;

Convergence Test

Set $U = \{U^1, U^2, U^3\}$
Test: $|||U - W||| : \varepsilon$?[1]
(No) → (****)
(Yes) → Graphing of results.

References

1. G. Baumann, F. Stenger, Fractional calculus and Sinc methods. Fract. Calc. Appl. Anal. **14**, 568–622 (2011)
2. G. Baumann, F. Stenger, Fractional adsorption diffusion. Fract. Calc. Appl. Anal. **16**, 737–764 (2013)
3. G. Baumann, F. Stenger, Sinc-approximations of fractional operators: a computing approach. Mathematics **3**(2), 444–480 (2015)
4. F.D. Gakhov, *Bounary Value Problems*, translated from Russian by I.N. Sneddon (Pergamon Press, Oxford, 1966)
5. L. Han, J. Xu, Proof of Stenger's conjecture on matrix $I^{(-1)}$ of Sinc methods. J. Comput. Appl. Math. **255**, 805–811 (2014)
6. N.I. Musskhelishvili, *Singular Integral Equations* (Noordhoff, Groningen, 1953)
7. F. Nöther, Über eine Klasse Singulären Integralgleichungen. Math. Ann. **82**, 42–63 (1921)
8. F. Stenger, *Numerical Methods Based on Sinc and Analytic Functions* (Springer, New York, 1993)
9. F. Stenger, Collocating convolutions. Math. Comput. **64**, 211–235 (1995)
10. F. Stenger, Polynomial function and derivative approximation of Sinc data. J. Complex. **25**, 292–302 (2009)
11. F. Stenger, *Handbook of Sinc Numerical Methods*. A 470-page Tutorial & Matlab Package (CRC Press, Boca Raton, 2011)
12. F. Stenger, A.R. Naghsh-Nilchi, J. Niebsch, R. Ramlau, Sampling methods for approximate solution of PDE, in *Inverse Problems, Image Analysis, and Medical Imaging*, ed. by Z. Nashed, O. Scherzer (American Mathematical Society, Providence, 2002), pp. 199–249

[1] An ordinary Frobeneous norm would suffice here.

Chapter 6
Sinc Convolution Examples

Abstract This chapter collects two examples demonstrating the application of the DFT-Sinc convolution algorithm. We apply the algorithm to an initial vector field with Gaussian and hyperbolic multiplier. The two multiplier functions used for initial conditions guarantee that natural Dirichlet boundary conditions are satisfied. The different asymptotic decays of the multiplier functions require a special choice of the step length. For both examples we indicate some practical applications.

6.1 Introduction

Before we discuss the numerical solution of the N–S equation let us comment on some symmetry properties of Eq. (1.1). The symmetries are useful for the interpretation of the derived numerical results. There are well-known symmetries of the classical Navier–Stokes equation for the three-dimensional incompressible viscous flow

$$\partial_t \boldsymbol{u} + \boldsymbol{u}\ \mathrm{div}(\boldsymbol{u}) = -\epsilon\ \Delta \boldsymbol{u} - \mathrm{grad}(p) \tag{6.1}$$

where ϵ is the viscosity of the fluid. The density is set to a constant $\rho = 1$ in (6.1). The system (6.1) is equivalent to (1.1), allowing for point symmetries. Point symmetries are determined by applying Lie's transformation theory to (6.1) [4–7, 9, 11, 12]. The idea behind Lie's approach is to use the invariance of the equations under symmetry transformations. The transformations are applied to independent variables (x, y, z, t) as well as to the dependent variables (u^1, u^2, u^3) and p. The laborious algebraic calculations nowadays can be performed by using computer algebra programs like *MathLie* [1, 2]. The symmetries under which (6.1) is invariant are Galilean transformations, rotations, and scale-dilatation of the independent and dependent variables. We verified the symmetries given in [4–7, 9, 11, 12] using *MathLie*. The benefit of a symmetry analysis is that the different types of solutions can be classified and some special solutions directly follow from such an analysis [4, 5, 7, 12]. For example, the Galilean invariance guarantees that

© Springer International Publishing AG 2016

F. Stenger et al., *Navier–Stokes Equations on* $\mathbb{R}^3 \times [0, T]$,
DOI 10.1007/978-3-319-27526-0_6

the N–S equation allows moving waves which are not our target for now. The scale-dilatation invariance given by

$$\bar{r} \to \lambda \bar{r}, t \to \lambda^2 t, \boldsymbol{u} \to \frac{1}{\lambda}\boldsymbol{u}, \text{ and } p \to \frac{1}{\lambda^2}p \tag{6.2}$$

with λ the scaling or group parameter results to similarity solutions [3, 6, 10]. Similarity solutions are particularly important for practical applications in fluid- and aerodynamics [3, 8, 13]. The solution type resulting from (6.2) possesses the analytic structure

$$\boldsymbol{u}(\bar{r}, t) = \frac{1}{\sqrt{\lambda}}\boldsymbol{u}\left(\frac{\bar{r}}{\sqrt{\lambda}}, \frac{t}{\lambda}\right) \text{ and } p(\bar{r}, t) = \frac{1}{\lambda}p\left(\frac{\bar{r}}{\lambda}, \frac{t}{\lambda}\right) \tag{6.3}$$

where $\boldsymbol{u}$ and p are solutions of the N–S equation. If the group parameter λ is chosen as $\lambda = t/T$ where T is a characteristic time in an integration process, we gain

$$\begin{aligned} \boldsymbol{u}(\bar{r}, t) &= \frac{\sqrt{T}}{\sqrt{t}}\boldsymbol{u}\left(\frac{\bar{r}\sqrt{T}}{\sqrt{t}}, T\right) = \frac{1}{\sqrt{\tau}}\boldsymbol{u}_T\left(\frac{\bar{r}}{\sqrt{\tau}}\right) \text{ and} \\ p(\bar{r}, t) &= \frac{T}{t}p\left(\frac{\bar{r}T}{t}, T\right) = \frac{1}{\tau}p_T\left(\frac{\bar{r}}{\tau}\right) \text{ with } \tau = t/T. \end{aligned} \tag{6.4}$$

where $\boldsymbol{u}_T$ and p_T are solutions of the N–S equation. The scaling structure in (6.4) tells us that the solutions can be appropriately scaled by using a characteristic time T. Thus selecting an appropriate interval $[0, T]$ guaranties that the structure of the solution will be well defined [14]. This is also proven in Chap. 3 specifically for our numeric approach. Especially the scaling property of time allows us to make an interpretation of our numerical results as well as a transformation to a selected time scale. In addition to these symmetries of the N–S equation a fluid dynamical system under some constraints shows isotropic properties. Isotropy of the pressure is a well-known phenomenon under certain physical conditions. The most famous one is Pascal's law.

In the following sections we illustrate the application of the DFT-Sinc convolution to the N–S equation using the theoretical background discussed in the previous Chaps. 1–5. The aim is to derive approximate solutions in 3-d for the Navier–Stokes initial boundary value problem (1.1)–(1.3). We illustrate the solutions of the integral equations on a finite cube imbedded in an infinite continuum.

The numerical computations were performed on different notebooks (PCs) with Intel processors Core™2 Duo, Core™i7, and Core™i7vPro. The computers were used either as a single machine or in a network of four machines operating on Windows 7 with at least 3 GB main memory. The code for our algorithms is written in *Mathematica* 10. A detailed description of the *Mathematica* functions is given in Appendix A.

The use of small scale computers implies that we were looking for an effective implementation using specifically parallel capabilities of *Mathematica*. The result is an implementation using sequential and parallel computations together. Whenever possible and suitable for the calculations the *Mathematica* functions adapt to the needs of the computation. This way the resources of computers like memory and speed of processors are automatically selected.

A spin off, of the DFT-Sinc convolution implementation is, we need not set up the "big matrices" that are required for the solution of PDE via finite difference and finite element methods. These big matrices are one of the reasons why it is not possible to achieve high accuracy in solving PDE via these classical methods. Another reason is that the rate of convergence of the DFT-Sinc convolution technique is much more rapid than standard methods offer [16].

To get an idea of the complexity of numerical work let us note the following estimation. In a typical calculation with $N = 20$ Sinc points – which is fairly large for DFT Sinc calculation – we find that we get at least 5 places of accuracy. The size of the corresponding matrix involved in the computations is $41^3 \times 41^3$, or 68921×68921 elements. Such a matrix, which is full, contains in total $4750104241 \approx 4.75 \times 10^9$ elements. If such a matrix were to be obtained by a Galerkin scheme, with each entry requiring the evaluation of a three-dimensional integral, and with each integral requiring 41^3 evaluation points, then more than 3.27375×10^{14} function evaluations would be required, an apocalyptic task indeed, lasting approximately 3 years on present–day computers. On the other hand, our method accurately gives us all of these values in minutes, and for relatively little work.

The implementation of the temporal-spatial convolution algorithm follows the outline of the main algorithm discussed in Chap. 5. The main target of the implementation in *Mathematica* was to combine symbolic and numeric calculations, the so-called hybrid calculations. The use of symbolic and numeric calculations opens the potential to manipulate some of the calculations symbolically without numerical assignments. Using such intermediate symbolic results can finally be turned into numerical procedures delivering numerical values. The numeric values in turn can be used again in a symbolic representation of the final results. This combination of numeric and symbolic calculation opens a broad field of application in the design of algorithms in general. For our specific problem it simplifies the calculations and allows a direct implementation of the mathematical definitions and theorems discussed in the previous Chaps. 1–5.

The specific implementation for our calculations is based on a single interface function **NavierStokes3D[]**. This main function uses the essentials of Eqs. (1.1)–(1.3) like initial conditions, domain ranges, and discretization data. The main function **NavierStokes3D[]** uses several sub-functions related to specific tasks needed for the different steps in the algorithm. Such sub-functions are assigned to discrete Sinc-Fourier transforms and its inverse, calculation of convolution integrals in 3-d, generation and application of conformal maps as well as their inverse, and generation of discrete Sinc grids in 3-d and along the time axis. The details of these sub-functions are discussed in Appendix A.

The use of *Mathematica* has thus enabled us to effectively unite the definitions, theorems and lemmas of the previous chapters.

6.2 Example 1: Gaussian Multiplier

Our first example is the same example used in [14]. We use this type of initial-boundary value problem as a reference for our results. Note the solution algorithm in [14] is different from the current one. The current implementation combines DFT and a spatial Sinc convolution while in [14] uses only a Sinc convolution approach in all coordinates was used. We start out with the same divergence free initial velocity vector field $\boldsymbol{u}^0$ given by

$$\boldsymbol{u}^0 = (y-z, z-x, x-y)e^{-r^2} \text{ with } r^2 = x^2+y^2+z^2. \tag{6.5}$$

The initial condition satisfies natural boundary conditions; i.e., $\lim_{r\to\infty} \boldsymbol{u} = 0$. Such kind of boundary conditions can be generated at an initial stage by choosing a bell-shaped multiplier function as a common factor in the vector field $\boldsymbol{u}^0$ on $\mathbb{R}^3$. In *Mathematica* the initial vector field $\boldsymbol{u}^0$ is represented by

$$\mathbf{u0} = \{y-z, z-x, x-y\}e^{-\{x,y,z\}.\{x,y,z\}}$$

$$\left\{e^{-x^2-y^2-z^2}(y-z), e^{-x^2-y^2-z^2}(-x+z), e^{-x^2-y^2-z^2}(x-y)\right\}$$

where we use *Mathematica*'s vector notation to collect the components of a vector field in a 3×1 matrix. The coordinate vector $\{x, y, z\}$ is used in the same manner to generate r^2 in e^{-r^2} by applying a dot product to this vector. Based on the initial and boundary conditions we will compare the derived results with the ones presented in [15]. Such comparison opened insights in different directions for improving the calculations and the algorithmic approach.

Our aim with this example is to present an overview of the numerical solutions. We also demonstrate the calculation of some properties like flow fields, pressure distribution, and vorticity important in practical applications. In addition to the results given in [15], we verify that the error decay of the calculations follows an exponential relation as the number of Sinc points increases. However, the calculated error presented is not an absolute error as estimated in Chaps. 2 and 3. Instead we use the contraction mapping result of Chap. 4 to derive a relative error which approaches the absolute error as $N \to \infty$. As proved in Chap. 4 the algorithm satisfies a contraction condition of a fixed point scenario that allows us to compare solutions relative to a reference that yields the expected decay.

At the outset, we first collect information already available by the initial condition. We deviate here from the procedure of [15], in order to get more insight into the physical properties of the solution of the PDE. The first derived information is related to the initial vector field itself. We demonstrate how the flow lines and their strength are distributed in the spatial domain $\mathscr{D} = \{(x, y, z) : -3 \le x \le 3, -3 \le$

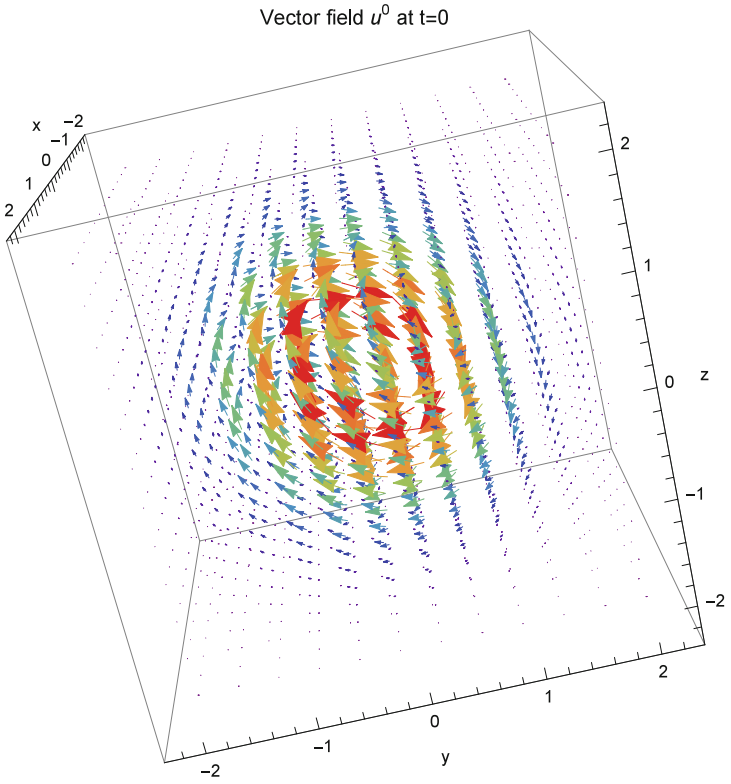

Fig. 6.1 Initial distribution of velocities in 3-d space around the origin. The length of the *arrows* represent the magnitude of the velocity field at that position

$y \leq 3, -3 \leq z \leq 3\}$ by graphing the initial vector field. A 3-d representation of the initial vector field is shown in Fig. 6.1. The graph shows that the initial vector field is exponentially decaying due to the Gaussian multiplier. The length of the vectors that depict the field strength decrease rapidly away from the origin, due to the Gaussian multiplier. In addition, we observe in Fig. 6.1 that there is a certain rotation of vectors present in the vector field $\boldsymbol{u}^0$.

To quantify the rotatory behavior of the vector field we calculate the vorticity $\boldsymbol{\omega}$ by applying the curl to $\boldsymbol{u}^0$ to find

$$\textbf{vorticity} = \textbf{Simplify}\left[\nabla_{\{x,y,z\}} \times \textbf{u0}\right]$$

$$\left\{2e^{-x^2-y^2-z^2}\left(-x(y+z)+y^2+z^2-1\right),\right.$$
$$2e^{-x^2-y^2-z^2}\left(x^2-xy-yz+z^2-1\right),$$
$$\left.2e^{-x^2-y^2-z^2}\left(x^2-xz+y^2-yz-1\right)\right\}$$

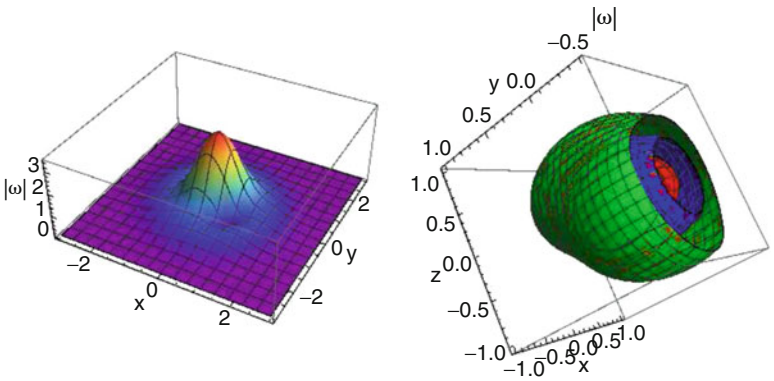

Fig. 6.2 Strength of the vorticity $|\boldsymbol{\omega}| = \left|\text{curl } \left(\boldsymbol{u}^0\right)\right|$ at $z = 0$

The result demonstrates that the vector field although divergence free, $\text{div } \left(\boldsymbol{u}^0\right) = 0$, possesses a certain rotational strength $|\boldsymbol{\omega}| = \left|\text{curl } \left(\boldsymbol{u}^0\right)\right|$. The vorticity strength $|\boldsymbol{\omega}|$ is shown in Fig. 6.2 where the spatial distribution on the domain $\mathscr{D}$ with $z = 0$ is visible (left panel). The vorticity strength $|\boldsymbol{\omega}|$ is also a bell-shaped function in this section of the space $(x, y, z = 0) \in \mathbb{R}^3$. From Fig. 6.2 we derive that the strength of vorticity is asymmetric around the origin. There are two dips along the off diagonal direction in the (x, y) plane. The right panel of Fig. 6.2 shows surfaces of constant vorticity strength; i.e., $|\boldsymbol{\omega}| = const$. The surfaces in $\mathbb{R}^3$ are distorted ellipsoids for lower values of $|\boldsymbol{\omega}|$ and become ellipsoids for larger values of $|\boldsymbol{\omega}|$. The vorticity and its strength are an essential property of the vector field shown in Fig. 6.1.

Since we are dealing with a 3-d flow it is difficult to follow all the details of the complex flow pattern (see Fig. 6.1). To get an overview of the behavior of the streaming fluid and the influence of the rotation, we use sections of the vector field $\boldsymbol{u}^0$ at specified locations along the z-axis. Thus we slice the 3-d continuum into a few pictures showing how the vector field and its strength are distributed. Figure 6.3 shows a sequences of slices along the z–axis. Figure 6.3 collects information about the projection of the vector field components $\left(u^{0,1}, u^{0,2}\right)$ in (x, y)-directions. Streamlines (circles) are used in this figure to depict the structure of the initial flow. In addition we graph as colored contours the strength of the total vector field $\boldsymbol{u}^0$ using the norm of the vector field as a measure. It is clear from these graphs that the larger values of the vector field are always centered around the origin (bright colors). On the other hand, the center of the streamlines is shifted if we step along the z-axis.

Another important property derived from the initial vector field is the initial pressure. We hasten to mention that the initial pressure is not used in the algorithm as an initial condition. However, once the initial conditions of the velocity field are given the pressure is defined by (1.13). The spatial distribution of the initial pressure allows us to estimate qualitatively the range of pressure values and the expected spatial distribution. This initial pressure strength indicates the distribution

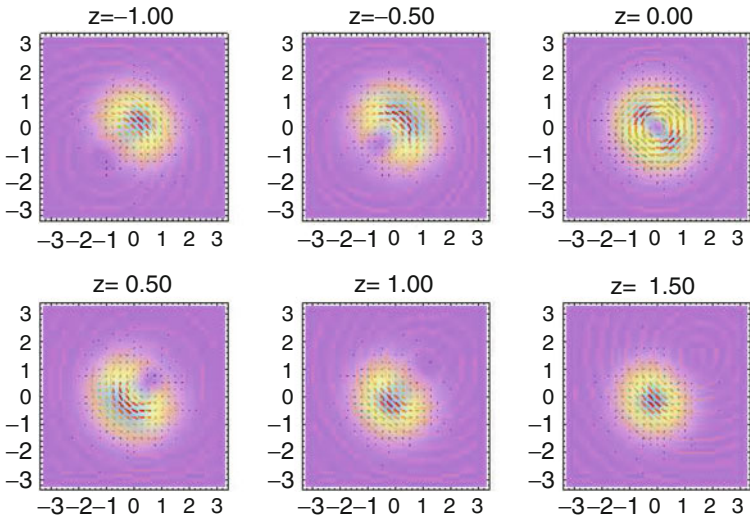

Fig. 6.3 Slices of the vector field at different positions of z (see *top* of graphs). Shown are the stream lines of a projection of the initial vector field (in x,y-components) and the vector field itself. The *colors* represent the local strength of the total vector field $\boldsymbol{u}^0$

of the expected pressure values in a Sinc convolution. Here we are only interested in the magnitude of pressure values. Once the initial conditions $\boldsymbol{u}^0$ are defined a potential function g can be derived using Eq. (1.14) of Theorem 1.4.1 (see also [8]). The implementation in *Mathematica* uses function **gh[]** for this purpose (see Appendix A). The function **gh[]** computes the potential via the formula

$$g(\bar{r}, t = 0) = \sum_{j=1}^{3} \sum_{k=1}^{3} u^j_{x^k}(\bar{r}, 0) u^k_{x^j}(\bar{r}, 0). \tag{6.6}$$

Formula (6.6) implemented in **gh[]** is used to derive the scalar function g applied to the previous definitions of $\boldsymbol{u}^0$ by

ggh = Simplify [gh (u0 , {x, y, z})]

$$e^{-2(x^2+y^2+z^2)} \left(-6 + 8x^2 + 8y^2 - 8yz + 8z^2 - 8x(y+z)\right)$$

Since the original vector field consists of a linear polynomial multiplied by a Gaussian, we get for g a quadratic polynomial in x, y, and z multiplied by the Gaussian. A graphical representation of this potential function g at a section with respect to the z–axis is shown in Fig. 6.4 (left panel). The function in Fig. 6.4 is again a bell-shaped function oriented in the negative direction with respect to the z–axis. In practical applications such as in weather forecasting it is of significant importance to know the spatial pressure distribution. Such information is conveyed to us as isobars

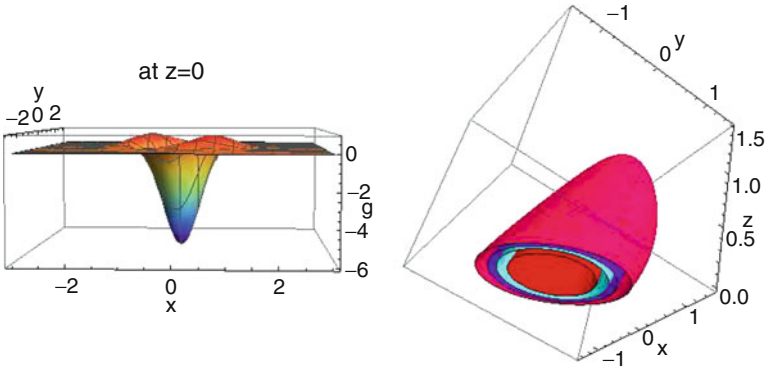

Fig. 6.4 Spatial representation of the function $g(\boldsymbol{r}, t = 0)$ based on initial conditions $\boldsymbol{u}^0$. The *left panel* shows the *bell-shaped* surface at $z = 0$. The right panel illustrates a section of the total geometric structure, showing the bell–shaped surface at $z = 0$

in daily weather forecasts. In the current approach the geometric structure of g is the source of such information. For the current calculation the geometric structure of g at constant values is a distorted ellipsoid. Different surfaces with constant negative values of g are shown in Fig. 6.4 right panel.

According to the discussion in Chap. 1 $g(\bar{r}, 0)$ is used in the following step to solve Poisson's equation (1.18) for generating the pressure. Possion's equation for the pressure is given as

$$\Delta p = -g \tag{6.7}$$

which is equivalent to the convolution representation (1.13) at $t = 0$

$$p = \int_{\mathbb{R}^3} G_0\left(\bar{r} - \bar{r}'\right) g\left(\bar{r}', 0\right) d\bar{r}'. \tag{6.8}$$

using $G_0(r) = 1/(4\pi r)$ as Green's function of the free space Poisson problem. We note that the pressure is calculated on $\mathbb{R}^3$ while later on the N–S equations are solved on $\mathscr{D}$.

The implementation of the pressure convolution integral (6.8) is included in the *Mathematica* function **convolutionIntegrals3DS[]** [14]. The function uses as input data the potential g, the domain $\mathbb{R}^3$, the number of Sinc points N, and a parameter d controlling the step length of the Sinc collocation. A calculation using the above defined potential ***ggh*** delivers the initial pressure $p(\boldsymbol{r}, 0)$ by applying the function **convolutionIntegrals3DS[]** to it

solP2S = convolutionIntegrals3DS [ggh , {x, −∞, ∞}, {y, −∞, ∞}, {z, −∞, ∞}, 4, 4, 4, 2];

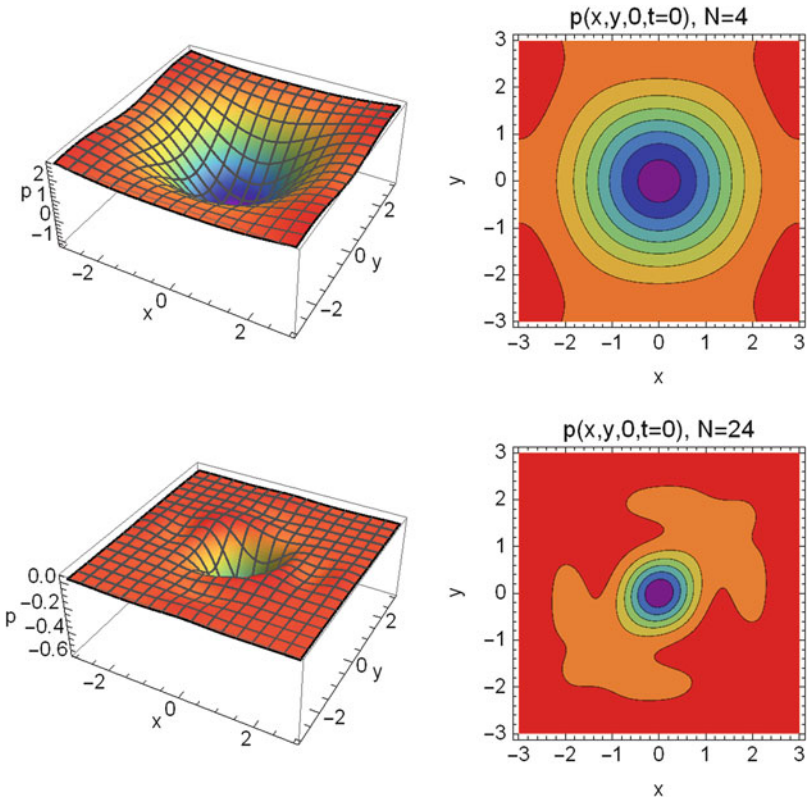

Fig. 6.5 Initial pressure distribution at a section with $z = 0$ as 3-d plot and contour plot showing the isobars in the fluid. The *top row* uses $N = 4$ Sinc points; the *bottom row* $N = 24$

It is important to note that the calculation of the convolution integral in (6.8) is carried out on the total space $\mathbb{R}^3$. In our implementation there are no numeric approximations of infinity. The result is a representation at the initial pressure in terms of a Sinc approximation which is shown in Fig. 6.5. We also note that the pressure approximation derived via the application of **convolutionIntegrals3DS[]** to the potential g is an analytic function in 3-d. The initial pressure shown in Fig. 6.5 has a similar bell-shaped structure as the potential function g. Due to the convolution of g with Green's function $G_0(r)$ the original bell-shaped function is distorted to a rectangular pyramidal shape at the base and rounded on the tip. This only holds for a Sinc approximation with a small number of approximation points; e.g., $N = 4$. The distortion is obvious in the right panel of Fig. 6.5 top row where we plotted contours of the pressure, the so-called isobar lines. Note that the ripples on the boundary are also due to the small number of Sinc points. If the number of Sinc points is increased ripples disappear and the rectangular base shape becomes an elliptic one

(see Fig. 6.5 bottom row). The sensitivity of the pressure on the number of Sinc points N is important for the interpretation of our later results.

Another quite useful source of information for judging on the derived approximation in a Sinc convolution is the vector field of the initial velocity. In Fig. 6.1 we represent the initial vector field of the velocity by a set of planar vectors. The different colors measure the strength of the local field (see Fig. 6.1). Since the flow field structure in 3-d is complicated a simplification is needed. For this reason we use the velocity components at specific sections to display the 3-d solution structure. Thus the derived information from calculations is more accessible if 3-d plots are used to represent the spatial distribution of the velocity. The method of representation is repeated for the different components and directions of the vector field. Such sections of the components for $\boldsymbol{u}^0$ are shown in Fig. 6.6. The set of initial conditions $\boldsymbol{u}^0$ can be graphically represented for a first overview of the solution by using the function **vecInitialValuesPlot[]**. The function uses the initial vector field $\boldsymbol{u}^0$, the domain $\mathscr{D}$, and three coordinate values for the sections of the 3-d vector field.

vecInitialValuesPlot [u0 , {x, −3, 3}, {y, −3, 3}, {z, −3, 3},

{$z \to 0, z \to 0, y \to 0$}]

The spatial distribution of the initial condition indicates how the approximated solution of the N–S system may look like in a Sinc approximation. It goes without

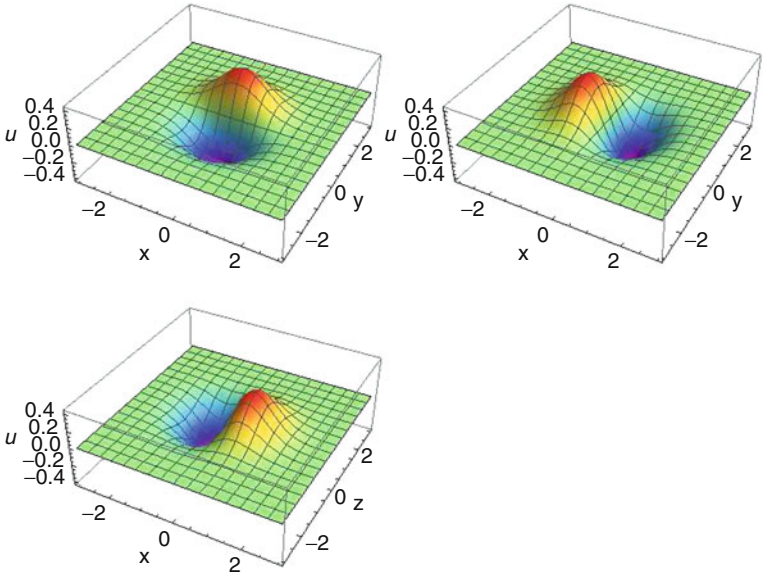

Fig. 6.6 Initial velocity components $u^{0,1}$, $u^{0,2}$, and $u^{0,3}$ at sections ($z = 0, z = 0, y = 0$) are shown in panels *left to right* and *top down*, respectively. The domain of representation is $\mathscr{D} = \{(x, y, z) : -3 \le x \le 3, -3 \le y \le 3, -3 \le z \le 3\}$

saying that the final amplitudes and variations will be different from the initial structure.

Having defined the initial and boundary properties, we are in a position to carry out a spatial-temporal approximation of the problem formulated in (1.1)–(1.3). For the symbolic and numerical approximation we will use a function called **NavierStokes3D[]** . For implementation details of the function see Appendix A.The main function **NavierStokes3D[]** uses the initial vector field $\boldsymbol{u}^0$, the domain $\mathscr{D} \times [0, T]$, the Sinc points N and M for spatial and temporal discretization, a step length parameter δ, viscosity ϵ, and an error margin to check the convergence of the Gauss–Seidel iteration. In addition to the necessary input data the function uses an option allowing specification a file name used to store all the information of input and output data generated.

We illustrate below our program layout that includes routines for computing **NavierStokes3D[]** as a function of the input data. The amount of data that is generated is quite large, and for sake of illustration, we attempted to limit this amount by limiting the number of Sinc points. The computations based on the Gauss–Seidel iteration is terminated by an error criterion that is computed by a Frobeneous norm.

Our experience regarding the convergence of the algorithm is the following. If one of the vector field iteration errors is larger than $1/2$, then the Gauss–Seidel iteration scheme fails to converge. If the error values are below this experimental threshold value of $1/2$ we have convergence of the iteration. We also note, if the errors are larger than this threshold value, we have to adjust the time interval to reach convergence. The adjustment of the time interval is not a problem because larger times T can be reached by chaining the calculations using the results of the previous one as initial conditions. This holds as long as the algorithm converges. Note, such an approach is in agreement with the proofs given in Chap. 4. We also mention that increasing the number of spatial and temporal Sinc points needs an adjustment of the time interval. The following line shows a typical example of a calculation to solve the coupled integral equations for the Navier–Stokes equations.

```
solutionsNS =

NavierStokes3D [u0 , {x, −3, 3}, {y, −3, 3}, {z, −3, 3}, {t, 0, 1. 10^−6} ,

2, 1, 2, .5, 10^ − 2, exportResultsTo → {" demonstrationExample_001.m"}]//

AbsoluteTiming
```

– Spectrum done —
– enter Loop —
– Fourier I done —
– convolution I —
— parallel time convol I = 22.538605
– convolution I done —
– time step I done —

– Fourier II done —
– convolution II —
– parallel time = 22.568609
– convolution II done —
– time step II done —
– Fourier III done —
– convolution III —
– parallel time = 22.224551
– convolution III done —
– time step III done —
erors ^{2} = {2.0042728487350446*^-9, 2.00427283593837*^-9,
2.004272823975107*^-9}
error sum = 0.0000775424

large output | show less | show more | show all | set size limit. . .

The results generated by **NavierStokes3D[]** are assigned to a list **solutionsNS**. This list consists of different elements of information. Above only a shortened version of results is displayed. The complete output file is listed in Appendix B.The elements of the generated list contain the following. The first entry in the list of results **solutionsNS** is the absolute time measured to derive the approximations. Here the total calculation lasted about 80 s. This entry is actually added to the list of results by applying the *Mathematica* function **AbsoluteTimiming[]** to the function **NavierStokes3D[]** and is not part of the results of **NavierStokes3D[]** in general. The essentials in the list of results are first the error components, second the analytic expressions for the velocity components u^1, u^2, u^3, and the pressure p. These major components use special symbols **uh1**, **uh2**, **uh3**, and **ph** to represent the functional structure. The symbol names mentioned are generated by **NavierStokes3D[]**. The resulting functions are analytic expressions in x, y, z, and t. The representation of the different components for $\boldsymbol{u}$ and p are based on *Mathematica*'s **Function[]** definition so that not only the analytic representation of the function is available but also derivations of this representation. The functional representation uses replacement rules so that the vector field components and the pressure are generally applicable to any kind of relations defined as expressions of the vector field $\boldsymbol{u}$ and the pressure p. Once we have derived these results we can use them in any expression defined for the vector field and the pressure.

The last element in the results list **solutionsNS** collects the discrete information of the vector field components and the pressure. These discrete representations can be used, if needed, to change the symbolic representation or to be used in additional discrete applications.

The overall structure of the list of results **solutionsNS** is thus given by the pattern

{errors , uh1 , uh2 , uh3 , ph , {U1 , U2 , U3 , *p*}}

representing the derived information from function **NavierStokes3D[]**.

The total information needed to set up the approximation and to perform the calculation as well as the results of the computations are collected in a file of results (see Appendix B).The results file is generated by **NavierStokes3D[]** automatically. The name assigned to the file is specified as option **exportResultsTo**.

If we need to retrieve such results for additional calculations, we will use the function **importResults[]** with the file name specified when the calculation was carried out. The definition of function **importResults[]** is listed in Appendix A.An example for retrieving data from the disc is shown in the next line. All the data previously stored are read into *Mathematica* and assigned to the variable **solutionsNS** according to the following line

solutionsNS = importResults [" demonstrationExample_001.m"];

After reloading the results the same structure of the previously generated list of results is available. The last element of **solutionsNS** contains the results. The first part of the list contains the input information required by **NavierStokes3D[]**. Having this information available we can easily perform the same calculation again by applying the function **NavierStokes3D[]** to this first part of **solutionsNS**. The used input data are for the above example

First [solutionsNS]

$$\left\{\left\{e^{-x^2-y^2-z^2}(y-z), e^{-x^2-y^2-z^2}(-x+z), e^{-x^2-y^2-z^2}(x-y)\right\},\right.$$
$$\left.\{x,-3,3\},\{y,-3,3\},\{z,-3,3\},\{t,0,1.\grave{}*\text{^}\text{-}6\},2,1,2,0.5,\tfrac{1}{100}\right\}$$

As a result, we get the initial values, the domains, the number of Sinc point N and M, the step length parameter d, the viscosity, and the error margin, respectively. The collection of the input data also makes it easy to change the starting values for a new or additional calculation by using a larger number of Sinc points, for example.

The last part of list **solutionsNS** is organized exactly as discussed above and can be directly used in additional calculations and graphical representations.

We now illustrate below differing aspects of the solutions to the Navier–Stokes problem as functions of the generated results described above. We first will compare approximations calculated using a low number of Sinc points with one of larger Sinc points. Let us first reload two different calculations differing in the number of Sinc points. The first step is to define the location of the data on disc.

SetDirectory [" ..\\Errors_CRC"]

.. \\Errors_CRC "

We then retrieve the available results from these files by reloading the results for calculations with $N = 4$ and $N = 18$ Sinc points

solutionsNS4 = importResults [" exampleCRCResultsN_4_0_003.m"];

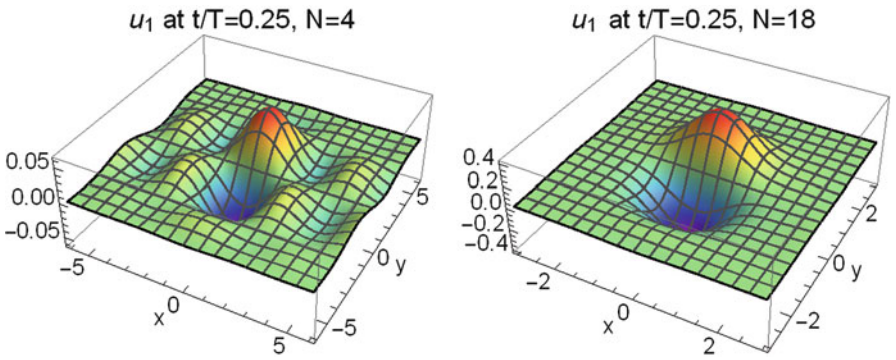

Fig. 6.7 Velocity component u^1 approximated with $N = 4$ and $N = 18$ Sinc points at $z = 0$ on the domain $\mathscr{D} = \{(x, y, z) : -3 \leq x \leq 3, -3 \leq y \leq 3, -3 \leq z \leq 3\}$. The solutions are generated at $t/T = 0.25$ using $M = 6$ Sinc points for the time domain

solutionsNS18 = importResults [" exampleCRCResultsN_18_0_011.m"];

As discussed the approximations for the different components are included in the last part of the two lists **solutionsNS4** and **solutionsNS18**. These results are used in the graphical representation of the first component of the vector field u^1 in Fig. 6.7. The generated graphs are arranged on a graphical grid in Fig. 6.7. The point is that we use the two specific solutions in replacement rules for u^1 at $z = 0$ at a fixed time $t/T = 0.25$.

The resulting graphs demonstrate that even for small numbers of Sinc points the structure of the solution is resolved. The approximation with $N = 4$ shows oscillations of the field u^1 especially at the boundaries. These variations disappear in the approximation with $N = 18$. It is also seen in Fig. 6.7 that an increase of N from 4 to 18 changes the steepness of the peak and valley. It is striking that the quality of the generated results with only a few array elements is highly accurate. For the rest of the presentation we will restrict ourselves to the case of $N = 18$ Sinc points.

We next demonstrate via a sequence of plots the static structure of the solutions at a fixed time t/T. In Fig. 6.8 we present the solutions for the different velocity components at $t/T = 0.25$ corresponding to $1/4$ of the total time interval.

The static solutions shown in Fig. 6.8 are similar to the original initial conditions. The representation of the vector field components in 3-d graphs is one way of getting an idea about the solution structure. However, in such graphs there is no direct link between the velocity components. This means that the graphs are displaying only separated parts of the real structure of a fluid motion. If we are interested in how the velocities especially the magnitude of the velocities are distributed in space, we can use the norm of the total vector field $\boldsymbol{u}$. Constant values of the norm define iso-velocity surfaces. To represent such spatial velocity distribution, we assume a constant value of the magnitude of the velocity and ask how this value is distributed in space. The type of graphs generated by such an approach are iso-velocity surfaces

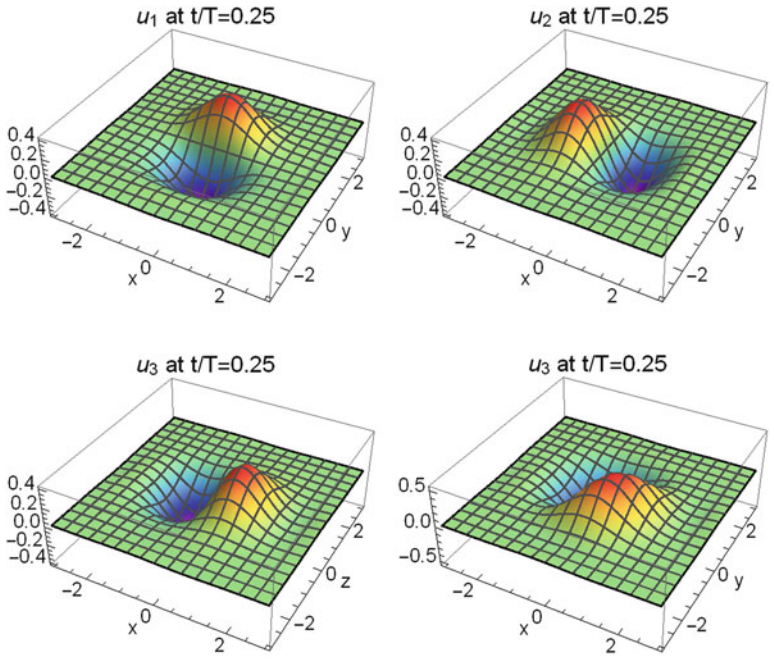

Fig. 6.8 Static velocity component u^1, u^2, and u^3 approximated with $N = 18$ Sinc points at $z = 0$ and $y = 0$ (*left bottom panel*) on the domain $\mathscr{D} = \{(x, y, z) : -3 \leq x \leq 3, -3 \leq y \leq 3, -3 \leq z \leq 3\}$ at $t/T = 0.25$ using $M = 6$ Sinc points on the time domain

in $\mathbb{R}^3$. In Fig. 6.9 we generated such surfaces of constant velocity for three different values. The structure of the graph shows that the surfaces of constant velocity are tori. In addition we observe a "pipe" connecting two regions of the surface by trespassing the center of the torus. The tori in Fig. 6.9 of constant velocity is determined for a fixed time $t/T = 0.25$. We note that the temporal component of the solution is still not included in our presentations. Note also that the directional information is lost in such a plot.

By including the temporal changes of the solution, we can take the approach to select one spatial coordinate and the time axis to represent a spatial-temporal structure. In addition we have to select two specific values for the remaining two spatial coordinates (complementary coordinates) to generate a 3-d graph. In Fig. 6.10 we collect graphs for different spatial directions. The two remaining coordinates not depicted in a graph were set to specific values and printed on top of the graphs. We observe in Fig. 6.10 that along the time axis there exists a sudden transition from zero to a finite value with only small variations. This finite value remains until the end of the interval. Suddenly at the end the amplitude approaches zero. This behavior is due to the selection of conformal maps for finite time intervals. Conformal maps for finite time intervals guarantee that the Sinc approximation satisfies Dirichlet boundary conditions. The maximum temporal

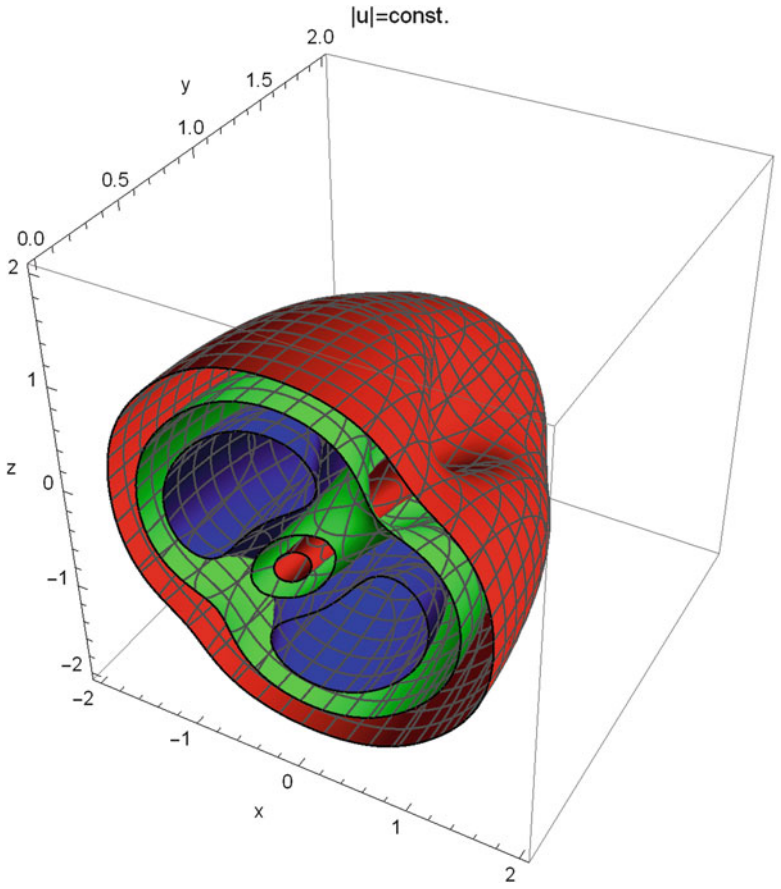

Fig. 6.9 Surfaces of constant velocity for the total vector field $\boldsymbol{u}$. The strength of $\boldsymbol{u}$ is determined by the norm and set to a constant value at a specific time $t/T = 0.25$. The three different surfaces represent three different total velocity strengths in the fluid. To uncover the structure of the nested tori we represent only a portion of the domain $\mathscr{D}$

value of the approximation stabilizes after a small initial time to a nearly constant value increasing again toward the end of the interval to a maximum. Finally this maximal value drops down to the original small value satisfying the boundary value. The maximums are rounded which may be a consequence of the number of Sinc points used along the time axis which is $M = 6$ in Fig. 6.10. To confirm this characteristic in the time domain we run the same calculation with $N = M = 8$ shown in Fig. 6.11. The overall structure with this higher resolution in time is the same but with a clear resolution of oscillations near the initial and terminal point in time.

In addition to the velocity vector field components we examined the temporal behavior of the pressure p along the different coordinate axis. The results are shown

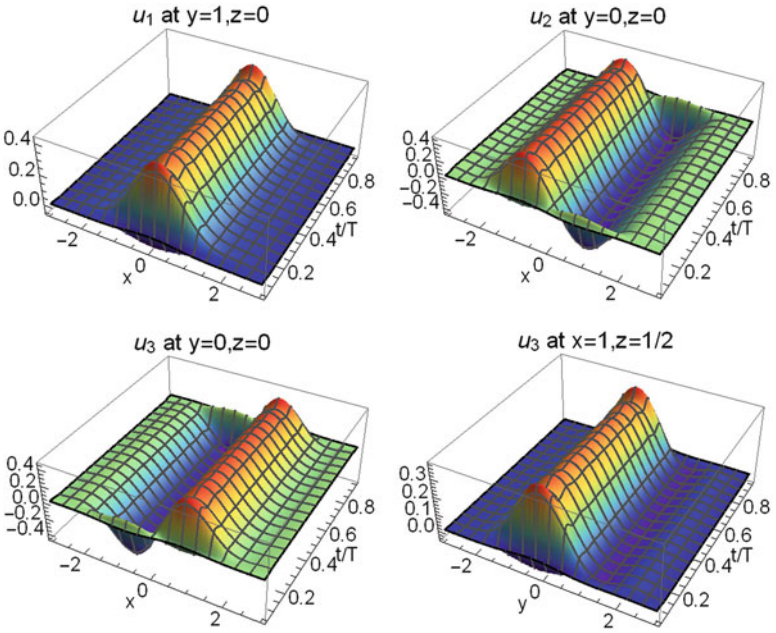

Fig. 6.10 Temporal-spatial representation of the velocity component u^1, u^2, and u^3 approximated with $N = 18$ Sinc points. The time interval was running from $[0, T]$ using $M = 6$ Sinc points. The graphs use different intersection locations of the complementary coordinates indicated at the *top* of each graph

in Fig. 6.12. We again set the complementary coordinates not depicted in the graphs equal to zero. It turns out that the temporal behavior in all coordinate directions under these conditions is the same. Thus the graphs in Fig. 6.12 show that the pressure is an isotropic quantity. Isotropy of the pressure must be expected because p is a scalar field which should not change its temporal behavior in different directions of the domain. However, the "temporal isotropy" is a special feature if we look at the spatial distribution of the pressure at a fixed time (see the panel in Fig. 6.12 bottom right). The spatial variation of the pressure is complex but resembles the initial pressure distribution shown in Fig. 6.5.

In Figs. 6.13 and 6.14 we represent isobar lines of constant pressure. As already mentioned the 3-d representation of the pressure at a fixed time resembles the initial pressure distribution but shows shifts of the low pressure center at different complementary locations. Such kind of shifts and locations of low pressure centers is shown in the left panels of Fig. 6.13.

In Fig. 6.14 we display the spatial distribution of constant pressure surfaces (isobar surfaces). The graph was generated for low constant pressure values around the center of the domain. These isobar surfaces have elliptic structure. Away from the center of the low pressure location these surfaces become distorted

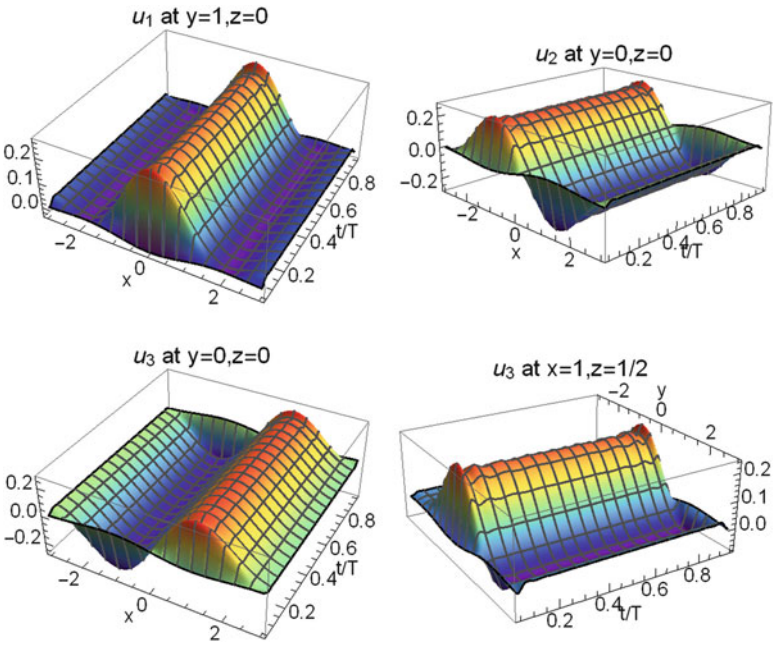

Fig. 6.11 Temporal-spatial representation of the velocity component u^1, u^2, and u^3 approximated with $N = 8$ Sinc points. The time interval was running from $[0, T]$ using $M = 8$. The graphs use different intersection locations of the complementary coordinates indicated at the *top* of each graph

ellipsoids. Far away from the low pressure center the surface becomes very complex corresponding to a pressure value which is nearly zero.

At the end of the example we pose the question of reliability. Specifically we are interested in how the error of our calculations decays. The theoretical prediction derived in Chap. 3 is that the errors decay exponentially. However, the question is how this exponential decay looks like for practical computations. To examine the question experimentally needs to repeat the calculations with different numbers of Sinc points. We repeated the calculation for the same initial vector field $\boldsymbol{u}^0$ for different numbers of Sinc points N to confirm the prediction of an exponentially decaying error of Sinc approximation as stated in Chap. 5. Since we are lacking of an exact analytic solution for the given initial vector field we rely on the validity of the contraction theorem of Chap. 4. Thus we are able to compare the calculated approximations with an "optimal" approximation with a large number of Sinc points. In our specific case we assume that we reach the optimal solution if $N = 18$. The derived errors are all with reference to this largest approximation. Since the choice of an optimal N is somehow arbitrary we also use reference approximations with smaller N. We start with the smallest $N = 8$ and increase the total number of Sinc points to $N = 18$. We note that all these calculations use a relative measure and can only indicate the type of error decay and will not deliver an absolute error.

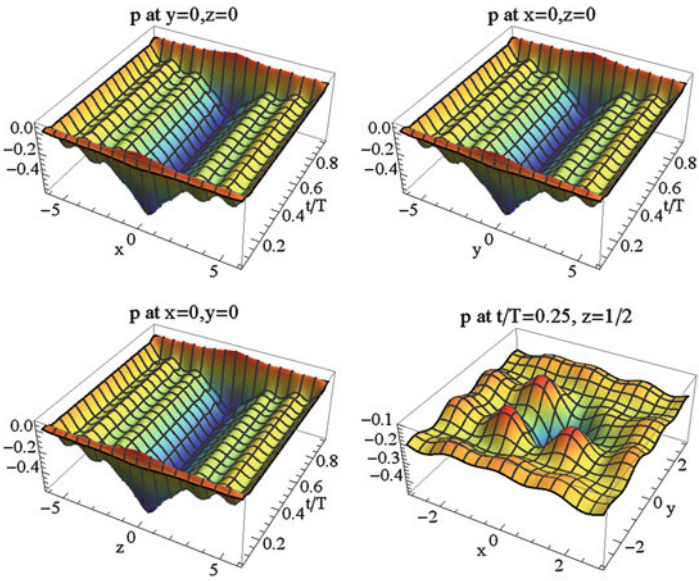

Fig. 6.12 Temporal and spatial distribution of pressure values p at specified complementary coordinate locations (see *top* of graphs). Time was running in the interval $[0, T]$ using $M = 6$ Sinc points. The *lower right panel* shows the spatial variation of the pressure p at $t/T = 0.25$

For the specific calculation of the error E_N^i we use the infinity norm to compare the lower Sinc point approximations with the largest one. We specifically use

$$E_N^i = \left\| u_j^i - u_N^i \right\|_\infty = \sup_{\mathbb{R}^3 \times [0,T]} \left(\left\| u_j^i - u_N^i \right\| \right) \text{ with } 8 \leq N \leq 18 \tag{6.9}$$

and $i = 1, 2, 3$, and $2 \leq j < N$. According to the theoretical prediction stated in Chap. 5 the error should follow an exponential decay as

$$E = aN^b e^{-cN^{1/2}} \tag{6.10}$$

where a, b, and c are constants independent of N. This phenomenological formula is motivated by the exact error relation $E_N \sim N^{1/2} \exp\left(-cN^{1/2}\right)$ for a Sinc error. To estimate the parameters a, b, and c in the phenomenological formula (6.10) we used a least square estimation for each E_N^i. The results of the least square estimations are given in Fig. 6.15. The related parameters are listed in Table 6.1. The observation is that increasing N as a reference the phenomenological relation (6.10) shows the tendency to approach the theoretical estimation. At least the estimated parameters stabilize for the larger values from $N = 12$ to $N = 18$.

In a second example we will use an initial condition which belongs to the space $A_{\alpha,d}^n$ showing the exact error formula for only a few numbers of Sinc points.

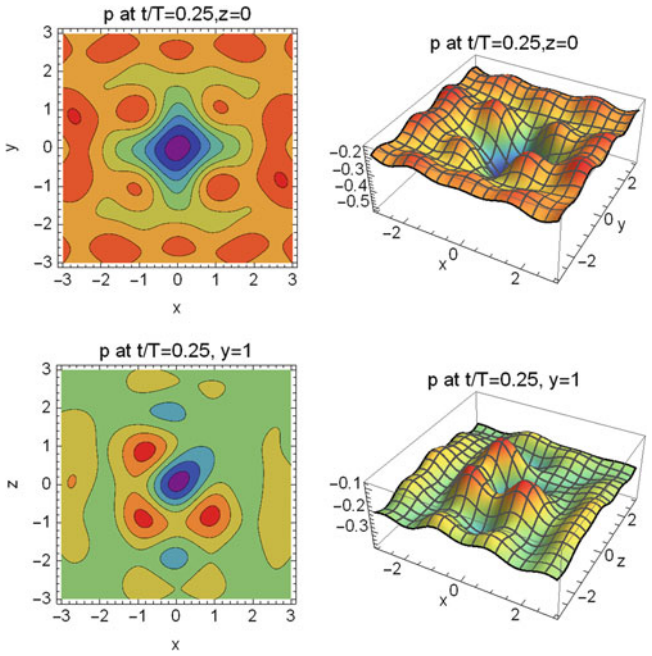

Fig. 6.13 Spatial distribution of the pressure p at a fixed time $t/T = 0.25$. The *left panels* show isobar lines as projections. The *right panels* show the 3-d sections of the pressure at different complementary locations (see *top* of the graph). The number of Sinc points used is $N = 18$ and $M = 6$

6.3 Example 2: Hyperbolic Multiplier

The second example deals with an initial vector field $\boldsymbol{u}^0$ belonging to the class $A_{\alpha,d}$ of functions dense in the class S of C^∞ functions which are rapidly decreasing. The generator of this class is of the type $1/\cosh(\cdot)$ which is used to set up the initial vector field $\boldsymbol{u}^0$ by

$$\boldsymbol{u}^0 = (y-z, z-x, x-y)/\cosh\left(\sqrt{\alpha\, r^2}\right) \text{ with } r^2 = x^2+y^2+z^2. \tag{6.11}$$

Compared with the previous example this initial condition differs only in the multiplier. The vector components of the velocity field are exactly the same. The common factor is a hyperbolic function instead of a Gaussian. The question we will examine is how does the change of the multiplier influence the solution structure and how are the convergence properties affected.

It becomes obvious from the comparison of the asymptotic behavior of the multipliers that the hyperbolic multiplier decays linearly with slope $\alpha^{1/2}$ and the Gaussian quadratically for large arguments $r \to \infty$. The asymptotic behavior of the multipliers is the main source of the differences which we will detect. The parameter

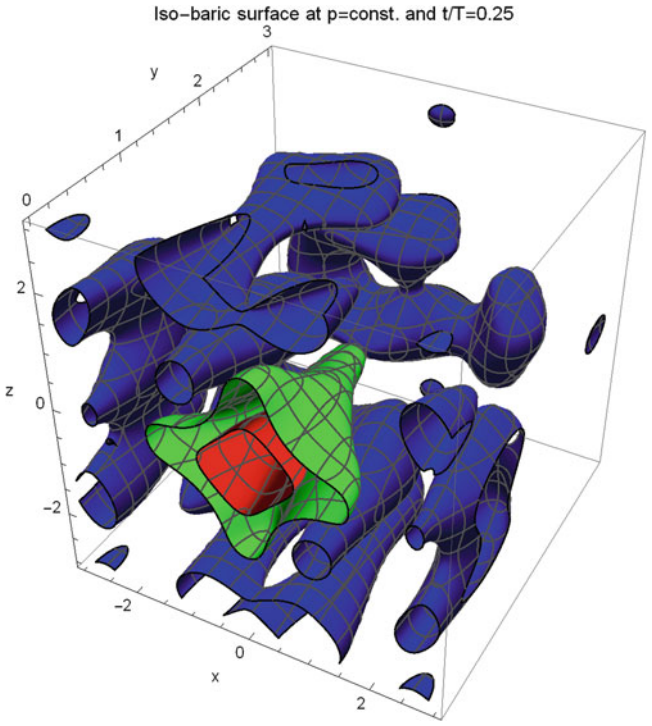

Fig. 6.14 Isobar surfaces around a low pressure center. Section of the total space at $y = 0$. The different surfaces correspond to different pressure values. The *elliptic shapes* represent negative small values. The fractions of surfaces are representing a pressure value near zero

α is important in choosing the step length and in achieving a certain convergence rate of the approximation. Thus we also expect an influence from this parameter.

Table 6.1 Least square parameters for E_N^i used in (6.10). For $8 \leq N \leq 18$ a subset of the data was used for the least square approximation

N	a	b	c
8	945.91	10.40	8.385
10	5849.72	12.954	10.867
12	72.212	7.523	6.207
14	89.100	8.011	6.745
16	100.486	8.149	6.857
18	82.677	7.646	6.271

The initial vector field satisfies the natural Dirichlet boundary conditions $\lim_{r\to\infty} \boldsymbol{u}^0 = 0$. In *Mathematica* the vector field is represented by the expression

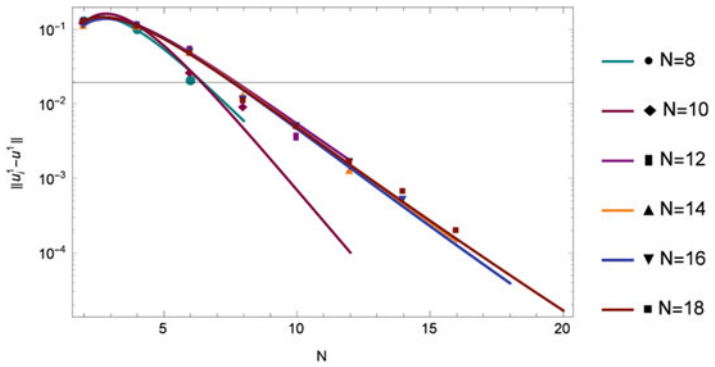

Fig. 6.15 Error decay of Sinc approximations for different Sinc numbers N. The number N indicated was the largest N used as reference

$$\mathbf{u0} = \frac{\{y-z, z-x, x-y\}}{\cosh\left(\sqrt{\alpha\{x,y,z\}.\{x,y,z\}}\right)} /. \alpha \to \frac{\pi}{4}$$

$$\left\{(y-z)\operatorname{sech}\left(\tfrac{1}{2}\sqrt{\pi}\sqrt{x^2+y^2+z^2}\right),\right.$$
$$(z-x)\operatorname{sech}\left(\tfrac{1}{2}\sqrt{\pi}\sqrt{x^2+y^2+z^2}\right),$$
$$\left.(x-y)\operatorname{sech}\left(\tfrac{1}{2}\sqrt{\pi}\sqrt{x^2+y^2+z^2}\right)\right\}$$

where we used $\alpha = \pi/4$ corresponding to the convergence rate of $\alpha d = \pi/2$.

If we examine the different properties of the initial condition as in the previous example, we now observe the same characteristics on a stretched scale. We exemplify this on the most striking similarities. The closest similarity shows up in the representation of surfaces of constant velocities. The iso-velocity surface for the above initial condition is depicted in Fig. 6.16. It is clear that again that a tori of the same structure is generated (see Fig. 6.9). There is, however, a small difference related to the geometric scale which is stretched in the current example.

The spatial distribution of the velocity components (see Fig. 6.17) shows a close similarity with the velocity components of the previous example (compare Fig. 6.8). The domain of the current example is due to the enlarged multiplier decay. We use here a typical domain $\mathscr{D} = \{(x, y, z) : -6 \le x \le 6, -6 \le y \le 6, -6 \le z \le 6\}$. The pressure p, in the lower right panel in Fig. 6.17, shows the same elliptic depression at the origin as before.

The spatial-temporal behavior of the solution is close to the spatial-temporal pattern in the previous example. The structure of the pressure related to the different coordinate axis is shown in Fig. 6.18. Again the scale is stretched for spatial coordinates. The same behavior is observed for the spatial-temporal behavior of the velocity components.

In our calculations we also checked the validity of Corollary 3.6.1, i.e. the isotropy integral of the pressure. We observe that for small numbers of spatial Sinc

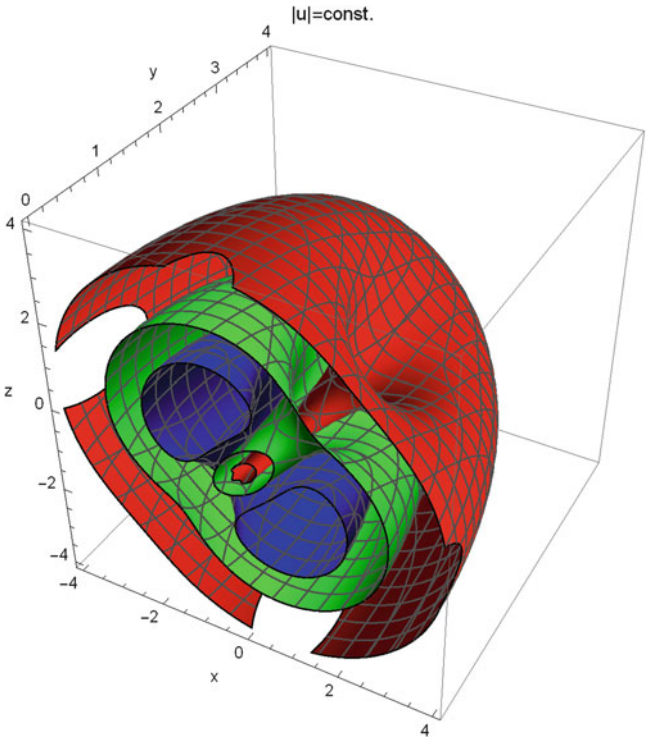

Fig. 6.16 Surfaces of constant velocity for the total vector field $\boldsymbol{u}$. The magnitude of $|\boldsymbol{u}|$ is determined by the norm and set to a constant value at a specific time $t/T = 0.25$. The three different surfaces represent three different total velocity values for the fluid. To uncover the structure of the nested tori we represent only a portion of the domain $\mathscr{D}$

points N the relation (3.47) is violated. However, if we increase the number of Sinc points N in our N–S approximation, then the relation (3.47), then the tends to zero. The approach to zero is depicted in Fig. 6.19. From Fig. 6.19 we can draw the conclusion that (3.47) is satisfied only asymptotically. Figure 6.19 shows the integral values on $\mathscr{D}$ for the two different multipliers of Examples 1 and 2. The calculations used $N = 12$ Sinc points in the 3-d Sinc quadrature. We also checked the influence of the number of Sinc points on the accuracy of the pressure integral, to deduce that the number of points required in our 3-d integrations increased to an unacceptable value. It became apparent that the result of Corollary 3.6.1 shows that the accuracy of our computations depends on the magnitude of the integral for the pressure.

Our algorithm for computing the components of $\mathbf{u} = (u^1, u^2, u^3)$ is carried out on a region

$$D_{M,N} = (-Nh, Nh)x(-Nh, Nh)x(-Nh, Nh)x(t_{-M}, t_M)$$

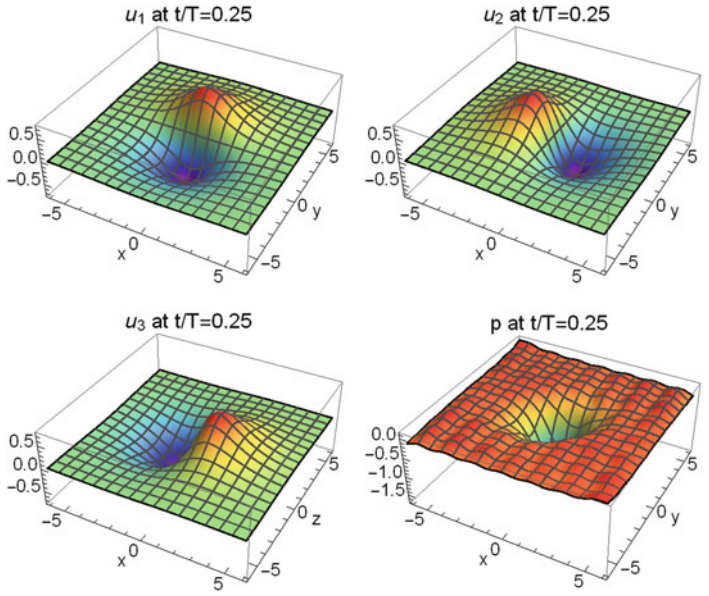

Fig. 6.17 Static velocity component u^1, u^2, and u^3 and the pressure p approximated with $N = 18$ Sinc points at $z = 0$ and $y = 0$ (*left bottom panel*) on the domain $\mathscr{D} = \{(x, y, z) : -6 \leq x \leq 6, -6 \leq y \leq 6, -6 \leq z \leq 6\}$ at $t/T = 0.25$ using $M = 6$ temporal Sinc points

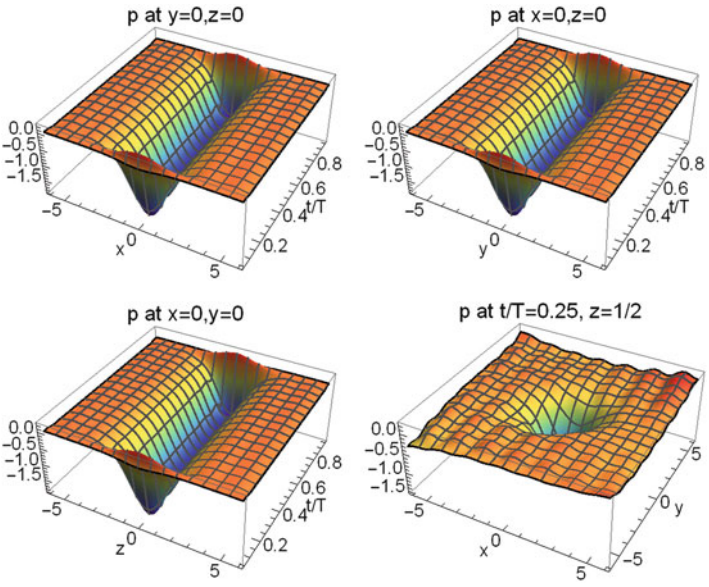

Fig. 6.18 Temporal and spatial distribution of pressure values p at specified complementary coordinate locations (see *top* of graphs). Time was running in the interval $[0, T]$ using $M = 6$ Sinc points. The *lower right panel* shows the spatial variation of the pressure p at $t/T = 0.25$

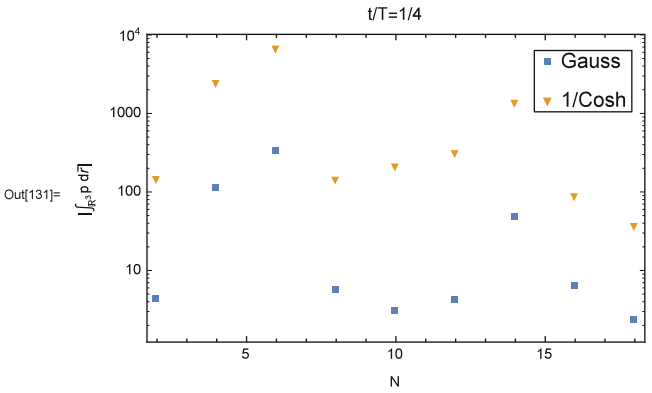

Fig. 6.19 Corollary 3.6.1 numerically verified. The Sinc-quadrature in 3-d used $N = 12$ Sinc points. The two sets of points are related to the different multipliers (see legend symbols)

whereas a large contribution of the pressure comes from the pressure integral outside of this region, i.e., from the region $\mathbb{R}^3 \setminus D_{M,N}$. At the moment we are unable verify this numerically.

6.4 Summary

The two examples demonstrate that the different multipliers affect the scaling of the approximation. The solution structure, however, is unaffected by the multipliers. We observe that in both cases the approximation structure of the solution is established for a small number of Sinc points. The velocity fields and their spatial-temporal representations are smooth surfaces in $\mathbb{R}^3 \times [0, T]$. The asymptotic behavior in spatial directions are fully present as natural Dirichlet boundary conditions. The pressure approximation, which is a derived quantity from the velocity fields, is sensitive in the spatial coordinates. Especially for small Sinc points N we observe ripples in the pressure approximations. The origin of these ripples is the spatial variations (first derivatives) of the velocity fields. The spatial sensitivity is reduced to a low level if the number of Sinc points N is increased. For values $N > 18$ the calculated approximations are acceptable.

References

1. G. Baumann, *Symmetry Analysis of Differential Equations Using Mathematica* (Springer, New York, 1998)
2. G. Baumann, Symmetry analysis of differential equations using *mathlie*. J. Math. Sci. **108**, 1052–1069 (2002)

3. G.W. Bluman, J.D. Cole, *Similarity Methods for Differential Equations* (Springer, New York, 1974)
4. V.O. Bytev, Invariant solutions of the Navier-Stokes equations. J. Appl. Mech. Tech. Phys. **13**, 809–816 (1972)
5. L.V. Kapitanskii, Group-theoretic analysis of the Navier-Stokes equations in the rotationally symmetric case and some new exact solutions. J. Math. Sci. **21**, 314–327 (1983)
6. S. Lie, G. Scheffers, *Vorlesungen über Differentialgleichungen mit Bekannten Infinitesimalen Transformationen* (B.G. Teubner, Leipzig, 1891)
7. S.P. Lloyd, The infinitesimal group of the Navier-Stokes equations. Acta Mech. **38**, 85–98 (1981)
8. A. Majda, A.L. Bertozzi, *Vorticity and Incompressible Flow* (Cambridge University Press, Cambridge, 2002)
9. S.V. Meleshko, V.V. Pukhnachev, One class of partially invariant solutions of the Navier-Stokes equations. J. Appl. Mech. Tech. Phys. **40**, 208–216 (1999)
10. P.J. Olver, *Applications of Lie Groups to Differential Equations* (Springer, New York, 1998)
11. V.V. Pukhnachev, Exact solutions of the hydrodynamic equations derived from partially invariant solutions. J. Appl. Mech. Tech. Phys. **44**, 317–323 (2003)
12. G. Rosen, Restricted invariance of the Navier-Stokes equation. Phys. Rev. A **22**, 313–314 (1980)
13. H. Schlichting, *Boundary-Layer Theory* (McGraw-Hill, New York, 1979)
14. F. Stenger, *Numerical Methods Based on Sinc and Analytic Functions* (Springer, New York, 1993)
15. F. Stenger, *Handbook of Sinc Numerical Methods*. Chapman and Hall/CRC Numerical Analysis and Scientific Computation Series (CRC Press, Boca Raton, 2011)
16. F. Stenger, A.R.N. Nilchi, J. Niebsch, R. Ramlau, Sampling methods for approximate solution of PDE. Contemp. Math. **313**, 199–250 (2002)

Appendix A
Implementation Notes

Abstract This Appendix collects explicit *Mathematica* functions for the solution of the N–S problem. We present a detailed implementation and a description of the functions. Examples demonstrate the application of several functions to typical input data. The functions are implemented in the *Mathematica Language* version 10. The algorithms and ideas presented in the previous chapters of this book are the basis for these function definitions. The main function **NavierStokes3D**[] collects all the mathematical definitions and theorems in a nut shell. The main function depends on different helper functions assigned to specific tasks like conformal maps, basis function generation, discrete Fourier transforms, generation of Sinc points, representation of Laplace transforms of a free space Green's function, the symbolic Laplace transform, convolution integrals, and input–output management of data.

A.1 Function Definitions

In the following subsections we discuss different functions which work as either symbolic or numeric functions. The added examples demonstrate how the *Mathematica* functions are used in connection with the main function **NavierStokes3D[]**. The implementation of *Mathematica* functions follow the mathematical definitions given in Chaps. 1–5. The *Mathematica* code is furnished with comments to increase the readability of the programs. Comments are included between the sequences $(* - - - \ldots - - - *)$. The comments are added whenever it is necessary to explain a specific property of the code.

The following sections are organized according to the topics outlined in the main algorithm introduced in Chap. 5. We start with the initialization process consisting of the setup and generation of initial values. A second set of functions creates helping functions like discrete Fourier transforms (DFTs), Laplace transform, and calculation of convolutions. The third set of helper functions is related to Sinc methods. We define the scaled and shifted Sinc function, indefinite integrals, convolution integrals, Green's function of the Poisson problem, conformal maps, and the basis function generation. The last set of *Mathematica* functions is devoted to the main algorithm of Chap. 5 extended by the input–output data management.

© Springer International Publishing AG 2016

F. Stenger et al., *Navier–Stokes Equations on* $\mathbb{R}^3 \times [0, T]$,
DOI 10.1007/978-3-319-27526-0

A.1.1 Selection of Initial Values

The calculations for solving the N–S equation are based on divergence free initial conditions. Thus it is beneficial to have a function in our tool box to generate such divergence free vector fields for a three component velocity field. To set up an effective way of calculation we first introduce a *Mathematica* function which generates a symbolic representation of such divergence free vector fields.

The function **initialValueSelection[]** is useful for generating an initial vector field $\boldsymbol{u}^0$ for the N–S problem. It uses the idea that any divergence free vector field can be generated by choosing two scalar functions v and w belonging to C^∞. Based on these two selected functions we create the initial vector $\boldsymbol{u}^0 = \left(u^{0,1}, u^{0,2}, u^{0,3}\right)$. To derive $\boldsymbol{u}^0$ from these two scalar functions several approaches are possible. One way to set up the vector $\boldsymbol{u}^0$ is to select two arbitrary analytic functions of 3 variables, v and w, and then to take $\mathbf{u}^0 = (v_z, w_z, -v_x - w_y)$. This approach is based on the fact that the *curl* of a vector field is gauge invariant. Based on these ideas it always follows $\text{div}\left(\boldsymbol{u}^0\right) = 0$ [7]. The *Mathematica* definition follows directly the outlined steps and collects the calculated vector field in a list. Lists in *Mathematica* allow the interpretation of vectors. The *Mathematica* definition follows next

$$\textbf{initialValueSelection[v_, w_]} := \left\{\partial_z v, \partial_z w, -\left(\partial_x v + \partial_y w\right)\right\}$$

The assumption in the definition above is that the coordinates of the functions v and w are Cartesian coordinates (x, y, z). We did not add a third argument to the function definition for coordinates because the current version of the algorithm is based on standard Cartesian coordinates. In the next steps we give two examples demonstrating the use of the function.

Example A.1.1. Divergence Free Vector Field

The application of this function **initialValueSelection[]** to two scalar fields shows how divergence free vector fields can be generated. By specifying the two functions v and w as input quantities in Cartesian coordinates (x, y, z), we derive the vector field. The following line gives an example of such derivation of a vector field

$$\textbf{vectorField} = \textbf{initialValueSelection}\left(x^2 z e^{-\left(x^2+y^2+z^2\right)}, y^2 z e^{-\left(x^2+y^2+z^2\right)}\right)$$

$$\left\{x^2 e^{-x^2-y^2-z^2} - 2x^2 z^2 e^{-x^2-y^2-z^2},\right.$$
$$y^2 e^{-x^2-y^2-z^2} - 2y^2 z^2 e^{-x^2-y^2-z^2},$$
$$\left.-2xze^{-x^2-y^2-z^2} - 2yze^{-x^2-y^2-z^2} + 2y^3 z e^{-x^2-y^2-z^2} + 2x^3 z e^{-x^2-y^2-z^2}\right\}$$

The check that this vector field is divergence free can be performed by the *Mathematica* function **Div[]**

$$\nabla_{\{x,y,z\}} \cdot \mathbf{vectorField}$$

0

The result demonstrates that the calculated vector field is in fact divergence free. ▽

Example A.1.2. Divergence Free Vector Field

If we discard the Gaussian multiplier in the previous Example A.1.1 we can directly verify the divergence free property of the vector field by a short inspection

$$\mathbf{vectorField2} = \mathbf{initialValueSelection}\left[zx^2, zy^2\right]$$

$$\left\{x^2, y^2, -2xz - 2yz\right\}$$

▽

Another quite useful tool for the generation of initial values is a function generating a graphical representation of the vector field components as 3-d plot. To represent a vector field with three components defined in Cartesian coordinates, we have to introduce constraints to the domain. Specifically we will use a two-dimensional subset of coordinates to generate a 3-d representation in a graph. To this end we slice the space by specifying a fixed value for one of the three coordinates. The remaining two coordinates are used then in a 3-d representation.

The next function **vecInitialValuesPlot**[] graphs the initial vector field $\boldsymbol{u}^0$ on a specified domain. The generated sections need an additional specification of the location for each component of the vector field. The domains are specified by lists consisting of the coordinate variable and the lower and upper bounds. The section level for the third coordinate is specified for each component by a specific value defined as a rule in *Mathematica*.

The first two components of the vector field $\boldsymbol{u}^0$ are represented by this function as (x, y) plots. The third component of $\boldsymbol{u}^0$ is generated as (x, z) plot. Consequently the first two section specifications are related to the z coordinate and the last to the y coordinate, respectively. The following lines show the definition using standard functions of *Mathematica*.

```
vecInitialValuesPlot[ vec_, { x_, ax_, bx_}, { y_, ay_, by_}, { z_, az_, bz_},
levelVector_] :=
GraphicsGrid[
{{ Plot3D[ Evaluate[ Part[ vec, 1] /. Part[ levelVector, 1]], {x, ax, bx},
{y, ay, by}, PlotRange → All, AxesLabel → { x, y, "u1"} ,
```

```
ColorFunction → "Rainbow", PlotPoints → 50],
Plot3D[ Evaluate[ Part[ vec, 2] /. Part[ levelVector, 2]], {x, ax, bx},
{y, ay, by}, PlotRange → All, AxesLabel → { x, y, "u2"} ,
ColorFunction → "Rainbow", PlotPoints → 50]},
{ Plot3D[ Evaluate[ Part[ vec, 3] /. Part[ levelVector, 3]], {x, ax, bx},
{z, az, bz}, PlotRange → All, AxesLabel → { x, z, "u3"} ,
ColorFunction → "Rainbow", PlotPoints → 50]}]
```

The next example demonstrates the application to a specific vector field.

Example A.1.3. Divergence Free Vector Field Plot

The application of **vecInitialValuesPlot**[] to the vector field generated in Example A.1.1 delivers three graphs. The graphs represent the three components of the vector field $\boldsymbol{u}^0$ in the domain $\mathscr{D} = \{(x, y, z) : -3 \leq x, y, z \leq 3\}$. The slice location is specified in the last argument of the function (Fig. A.1). The next line shows the use of the function

```
vecInitialValuesPlot[ vectorField, {x, −3, 3}, {y, −3, 3}, {z, −3, 3},
{z → 0, z → 1, y → 1}]
```

▽

A.1.2 Pressure Potential g

The pressure p used as gradient in the N–S equation is also a divergence free quantity. The calculation is based on the existence of a potential function g [see Eq. (1.14)]. The potential g is derived from the velocity field. The specific relation is introduced in Theorem 1.4.1. In turn this results to the solution of Poisson's equation in terms of a convolution integral; see Eq. (1.13). Using Green's function of the free space problem and the potential g allows to derive the divergence free pressure p. To initialize such kind of pressure calculations we first need a *Mathematica* function which generates the potential function g. This scalar function g is according to (1.14) defined as sum of products of first order derivatives with respect to all coordinates and all components of the velocity vector field as follows

$$g(\boldsymbol{r}, t) = \sum_{j=1}^{3} \sum_{k=1}^{3} u^j_{x^k}(\boldsymbol{r}, t) u^k_{x^j}(\boldsymbol{r}, t). \tag{A.1}$$

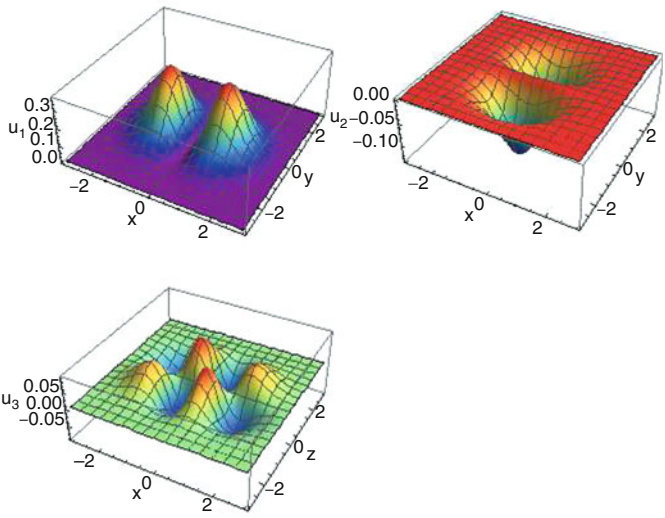

Fig. A.1 The graph shows the three components of a velocity vector field at slices $z = 0$ (*top row*) and $y = 1$

The double summation over coordinates and components is implemented in the *Mathematica* function **gh[]**. The input of the function is a vector field of velocities and the set of independent spatial variables. The symbolic result generated by **gh[]** is the scalar function $g : \mathbb{R}^3 \to \mathbb{R}$ representing the potential for the pressure p satisfying $\Delta \mathrm{p} = -g$. Poisson's equation is equivalent to a convolution integral using the potential g and the free space Green's function G in $p = \int_{\mathbb{R}^3} G(r - r')g(r')dr'$ (see Chaps. 1 and 3 for details). The explicit implementation of g follows next.

$$\mathbf{gh(\ uVec_,\ indepnd_)} := \sum_{j=1}^{\mathbf{Length[\ indepnd]}} \left(\sum_{k=1}^{\mathbf{Length[\ indepnd]}} \frac{\partial\ \mathbf{uVec[[}j\mathbf{]]}}{\partial\ \mathbf{indepnd[[}k\mathbf{]]}} \frac{\partial\ \mathbf{uVec[[}k\mathbf{]]}}{\partial\ \mathbf{indepnd[[}j\mathbf{]]}} \right)$$

The definition uses directly the definition introduced in Eq. (1.14). The symbol **uVec** represents a three-dimensional vector field of velocities depending on Cartesian coordinates.

Example A.1.4. Initial Pressure

Since the pressure equation is valid for all times we can specifically find the initial pressure using the initial velocity vector field $\boldsymbol{u}^0$ of Example A.1.1. To this end we need to calculate the potential g for the initial vector field. The first argument of **gh**[] is representing the vector field and the second one is a list of coordinate variables as given in the next line

gh(vectorField, {x, y, z}) // Simplify

$$8e^{-2\left(x^2+y^2+z^2\right)}\left(x^6\left(1+2z^2\right)+x^3y\left(-1+2y^2\right)\left(1+2z^2\right)+\right.$$
$$x^4\left(-2-7z^2+2z^4\right)+x^2\left(1-z^2+2z^4\right)-xy\left(-\left(1-2z^2\right)^2+y^2\left(1+2z^2\right)\right)+$$
$$y^2\left(1-z^2+2z^4+y^4\left(1+2z^2\right)+\right.$$
$$\left.\left.y^2\left(-2-7z^2+2z^4\right)\right)\right)$$

The resulting symbolic expression is a polynomial in variables (x, y, z) screened by the Gaussian multiplier.
▽

A.1.3 Time Steps

To initialize the N–S algorithm we need to define a set of mesh points on the time axis. This finite set of discrete points is generated by the inverse conformal map on a finite interval $[0, T]$. The following *Mathematica* function **tμ[]** generates the M Sinc point mesh on the interval $[0, T]$. The function uses the inverse conformal map $\psi_{a,b}(x)$ for finite intervals [see definition (A.10)]. The step length h_T is calculated internally as $h_T = \pi \Big/ \sqrt{M}$. The next few lines represent the function definition.

```
tμ[ T_, m_] := Block[{ hT},
(* -- calculation of step length h_T -- *)
hT = π / Sqrt[m] ;
(* -- generation of Sinc points -- *)
N[ ParallelTable[ T Exp[μ hT] / (1+ Exp[μ hT]), {μ, -m, m}]]]
```

The function uses as input data the end point T of the interval $[0, T]$ and the number of Sinc points M. The inverse conformal map is directly implemented for the finite interval in the last line of the definition above.

Example A.1.5. Sinc Points on $[0, T]$

The following example generates a list of Sinc points with $M = 5$ on the time interval $[0, 1]$.

```
tμ[1, 5]
```

```
{0.000888742, 0.00361208, 0.0145589, 0.0567902, 0.19703, 0.5,
0.80297, 0.94321, 0.985441, 0.996388, 0.999111}
```

The total number of generated Sinc points is $m = 2M + 1 = 11$. According to the conformal map in use the discrete Sinc points are nonuniformly distributed on the interval $[0, T]$.

▽

Note, the function **tμ[]** is useful only for numeric calculations.

A.1.4 Spatial Discretization

In addition to the temporal initial mesh the N–S algorithm also needs a spatial initial mesh for all Cartesian coordinates. Since the initial condition of the velocity vector field is given as a continuous function in Cartesian coordinates we need a transformation from the continuous (symbolic) representation to a discrete numeric one. This transformation is carried out in two steps. The first step is a discrete symbolic representation which is converted in the second step to a discrete numeric representation. The separation into a symbolic and numeric step becomes beneficial specifically in calculating the Cartesian product for the three components and three coordinates. This way the calculation can be reduced to a single outer product of one and the same discrete index list.

The function **u0Discrete[]** converts the symbolic initial vector field $\boldsymbol{u}^0$ to a symbolic discrete one. The function selects a single component of the vector field specified by the symbol **dimSelect**. The selected component is then transformed to a discrete symbolic representation by replacing the continuous variables by an index variable and the step length. This set must be repeated for all coordinates. The vector field $\boldsymbol{u}^0$ must thus be a function of the Cartesian coordinates (x, y, z). The selected component is then transformed to a symbolic discrete representation using indices j, k, and l, and a step length h. All of these quantities are input variables of the function. The function is useful for symbolic as well as for numeric calculations. The next lines list these definition in *Mathematica*.

```
u0Discrete[ u_List, dimSelect_, j_, k_, l_, h_] := Block[{ uselect},
(* -- select the component specified with dimSelect -- *)
uselect = Part[u, dimSelect];
(* -- continuous variables are replaced by discrete one -- *)
uselect /.{x → jh, y → kh, z → lh}]
```

The next example converts the vector field of Example A.1.1 to a discrete symbolic representation.

Example A.1.6. Discrete Symbolic Transform

The discrete symbolic representation for each component is generated as a table (list). The running index **component** selects from the vector field component one to three and replaces the Cartesian coordinates by an indexed one using a common step length h

Table[u0Discrete[vectorField, component, j, k, l, h], { component, 1, 3}]

$$\Big\{e^{-h^2j^2-h^2k^2-h^2l^2}h^2j^2 - 2e^{-h^2j^2-h^2k^2-h^2l^2}h^4j^2l^2,$$
$$e^{-h^2j^2-h^2k^2-h^2l^2}h^2k^2 - 2e^{-h^2j^2-h^2k^2-h^2l^2}h^4k^2l^2,$$
$$-2e^{-h^2j^2-h^2k^2-h^2l^2}h^2jl + 2e^{-h^2j^2-h^2k^2-h^2l^2}h^4j^3l -$$
$$2e^{-h^2j^2-h^2k^2-h^2l^2}h^2kl + 2e^{-h^2j^2-h^2k^2-h^2l^2}h^4k^3l\Big\}$$

We used the symbols j, k, l as index symbols. The result is a discrete representation of the initial vector field in a symbolic way. ▽

Example A.1.7. Numeric Transform

The numeric representation of the vector field needs numeric values for j, k, l, and h as shown next. We use the third component of the vector field to get the single value

u0Discrete[vectorField, 3, 1, 2, 3, .7]

0.00434817

▽

The function **u0Discrete[]** is delivering only one component discretized. However, we need in a numeric calculation all components. Such kind of discrete representation is generated by the following function **discreteInitialValue[]**. In addition, this function generates a special representation specifically useful for DFTs dealing with the half values at the boundary of the domain $\mathscr{D}$.

The function **discreteInitialValue[]** uses the function **u0Discrete[]** to generate a specific discretization at Sinc points for a specified component of the vector field. The selection of the component is based on the variable **dimSelect** using the values 1, 2, or 3. N specifies the number of Sinc points and h is the finite step length. The generated Sinc data are divided by 2 at the end points of the intervals. As mentioned in Chap. 5 this step is needed in the DFT later on. The following lines show the *Mathematica* definition of the function **discreteInitialValue[]**.

```
discreteInitialValue[ u0_, dimSelect_, n_, h_] := Block[{ indexList, tab1},
(* -- generation of an index list -- *)
indexList = Table[j, {j, -n, n}];
```

```
(* -- all permutation of indices are generated by an outer
product using the index list -- *)
tab1 =
Chop[N[ Outer[ u0Discrete[ u0, dimSelect, #1, #2, #3, h]&, indexList,
indexList, indexList]]];
(* -- adaption of the endpoints to halve the value -- *)
ParallelTable[ If[j == -n||k == -n||l == -n||j == n||k == n||l == n,
Part[ tab1, j + n + 1, k + n + 1, l + n + 1]/2,
Part[ tab1, j + n + 1, k + n + 1, l + n + 1]], {j, -n, n}, {k, -n, n}, {l, -n, n}]]
```

Example A.1.8. Discrete Initial Values

The application of the function **discreteInitialValue[]** to the vector field **vectorField** introduced in Example A.1.1 delivers an array of size $5 \times 5 \times 5$. We here used the third component with $N = 2$ Sinc points and a step length $h = 1/2$.

```
dU03 = discreteInitialValue[ vectorField, 3, 2, 1/2]
```

```
{{{0, 0, 0, 0, 0}, {-0.0395247, -0.0418369, 0, 0.0418369, 0.0395247}, {0, 0, 0, 0, 0},
{0.0395247, 0.0418369, 0, -0.0418369, -0.0395247}, {0, 0, 0, 0, 0}},
{{-0.0395247, -0.0418369, 0, 0.0418369, 0.0395247},
{-0.167348, -0.354275, 0, 0.354275, 0.167348},
{-0.107439, -0.227449, 0, 0.227449, 0.107439},
{0, 0, 0, 0, 0}, {-0.0395247, -0.0418369, 0, 0.0418369, 0.0395247}},
{{0, 0, 0, 0, 0},
{-0.107439, -0.227449, 0, 0.227449, 0.107439}, {0, 0, 0, 0, 0},
{0.107439, 0.227449, 0, -0.227449, -0.107439}, {0, 0, 0, 0, 0}},
{{0.0395247, 0.0418369, 0, -0.0418369, -0.0395247}, {0, 0, 0, 0, 0},
{0.107439, 0.227449, 0, -0.227449, -0.107439},
{0.167348, 0.354275, 0, -0.354275, -0.167348},
```

$\{0.0395247, 0.0418369, 0, -0.0418369, -0.0395247\}\},$

$\{\{0, 0, 0, 0, 0\}, \{-0.0395247, -0.0418369, 0, 0.0418369, 0.0395247\},$

$\{0, 0, 0, 0, 0\}, \{0.0395247, 0.0418369, 0, -0.0418369, -0.0395247\}, \{0, 0, 0, 0, 0\}\}\}$

$\triangledown$

A.1.5 Step Lengths Distribution

The N–S algorithm uses a specific definition of step length distribution defined in Eq. (5.33). This array of step lengths is frequently used in the algorithm. The array collects a weighted step length for each of the three indices j, k, l running from $-N \leq j, k, l \leq N$.

The corresponding *Mathematica* function **Λ[]** is generating the geometric discrete step length distribution for all j, k, and l using the number of Sinc points N and the common step length h as input data. The listing follows in the next line

$$\mathbf{\Lambda[\ n_,\ h_] := ParallelTable}\left[h\sqrt{j^2+k^2+l^2}, \{j, -n, n\}, \{k, -n, n\}, \{l, -n, n\}\right]$$

The function creates an array of dimension $(2N + 1)^3$ containing the different discretization lengths.

Example A.1.9. Discrete Step Length Distribution
An example is generated in the next line with $N = 1$ and $h = 1/3$

$$\mathbf{\Lambda}\left[1, \tfrac{1}{3}\right]$$

$$\left\{\left\{\left\{\tfrac{1}{\sqrt{3}}, \tfrac{\sqrt{2}}{3}, \tfrac{1}{\sqrt{3}}\right\}, \left\{\tfrac{\sqrt{2}}{3}, \tfrac{1}{3}, \tfrac{\sqrt{2}}{3}\right\}, \left\{\tfrac{1}{\sqrt{3}}, \tfrac{\sqrt{2}}{3}, \tfrac{1}{\sqrt{3}}\right\}\right\},\right.$$
$$\left\{\left\{\tfrac{\sqrt{2}}{3}, \tfrac{1}{3}, \tfrac{\sqrt{2}}{3}\right\}, \left\{\tfrac{1}{3}, 0, \tfrac{1}{3}\right\}, \left\{\tfrac{\sqrt{2}}{3}, \tfrac{1}{3}, \tfrac{\sqrt{2}}{3}\right\}\right\},$$
$$\left.\left\{\left\{\tfrac{1}{\sqrt{3}}, \tfrac{\sqrt{2}}{3}, \tfrac{1}{\sqrt{3}}\right\}, \left\{\tfrac{\sqrt{2}}{3}, \tfrac{1}{3}, \tfrac{\sqrt{2}}{3}\right\}, \left\{\tfrac{1}{\sqrt{3}}, \tfrac{\sqrt{2}}{3}, \tfrac{1}{\sqrt{3}}\right\}\right\}\right\}$$

The result is a $3 \times 3 \times 3$ array with different step lengths at the entries. $\triangledown$

A.1.6 Discrete Fourier Transform

One of the essential properties of the current implementation of the N–S algorithm is the use of DFT. The use of the DFT allows replacement of a 4-d convolution integral in spatial and temporal coordinates by a 3-d spatial convolution integral. The reduction of the integral dimension simplifies some of the calculations and speeds up the processing. DFT is basically used in the time developing steps of

the algorithm. DFT defined in [7] for a single variable is used here in the extension to three dimensions. Contrary to its definition as a matrix operation [7], we use a parallel implementation in 3-d which turns out to be faster than the related matrix product implementation.

The function **a3DFourier[]** contains the 3-d implementation of DFT and calculates a 3-d DFT using Sinc points. The 3-d DFT is basically represented by a triple sum which can be separated into three calculation steps as follows

$$\begin{aligned}\hat{f}(\xi,\psi,\zeta) &= h^3 \sum_{j=-N}^{N}\sum_{k=-N}^{N}\sum_{l=-N}^{N} f(jh,kh,lh)e^{ij\xi\frac{\pi}{N}}e^{ik\psi\frac{\pi}{N}}e^{il\zeta\frac{\pi}{N}} \\ &= h\sum_{j=-N}^{N} e^{ij\xi\frac{\pi}{N}} h\sum_{k=-N}^{N} e^{ik\psi\frac{\pi}{N}} h\sum_{l=-N}^{N} e^{il\zeta\frac{\pi}{N}} f(jh,kh,lh) \\ &= h\sum_{j=-N}^{N} e^{ij\xi\frac{\pi}{N}} h\sum_{k=-N}^{N} e^{ik\psi\frac{\pi}{N}} g(jh,kh,\zeta) \\ &= h\sum_{j=-N}^{N} e^{ij\xi\frac{\pi}{N}} w(jh,\psi,\zeta) \end{aligned} \tag{A.2}$$

The product of exponential functions is separated into three chained sums. These sums are independent of each other. Thus the summation over a single index can be performed independently from the others. The external indices of the discrete array can thus be run in parallel. This allows a break up of the triple product into three independent sums and a parallel set of framing indices. The speed up of the entanglement of the sums and the external frame indices delivers a factor 10 in performance. We note that an implementation using only a single **ParallelTable[]** and the three sums included in the framing table needs about ten times longer than the break up to three separated parallel tables. The function uses a list of equidistant data points assigned to the symbol **data**. The data points are separated from each other by the step length h. The total number of points is $m = 2N + 1$. The following lines define the DFT based on Sinc points. The implementation follows the outline of Eq. (A.2)

```
a3DFourier[ data_List, h_, n_] := Block[{ v1, v2, v3},
(* -- DFT of first coordinate -- *)
v1 = ParallelTable [ Sum [h Part[ data, j + n + 1, k + n + 1, l + n + 1] e^(i l λ π/n),
{l, -n, n}], {j, -n, n}, {k, -n, n}, {λ, -n, n}];
(* -- DFT of second coordinate -- *)
v2 = ParallelTable [ Sum [h Part[ v1, j + n + 1, k + n + 1, λ + n + 1] e^(i k η π/n),
```

```
{k, −n, n}], {j, −n, n}, {η, −n, n}, {λ, −n, n}];
(* −− DFT of third coordinate −− *)
v3 = ParallelTable[ Sum[h Part[ v2, j + n + 1, η + n + 1, λ + n + 1]e^(ijξ π/n),
{j, −n, n}], {ξ, −n, n}, {η, −n, n}, {λ, −n, n}];
Chop[ v3]
]
```

The function **inverseB3DFourier[]** calculates a 3-d discrete inverse Fourier transform using Sinc points. Similar to the forward DFT function **a3DFourier[]** for the inverse DFT, we split up the 3-d sum in three single summations with a parallel frame of indices j, k, l.

$$
\begin{aligned}
f(j,k,l) &= \frac{1}{(2Nh)^3} \sum_{\xi=-N}^{N} \sum_{\psi=-N}^{N} \sum_{\zeta=-N}^{N} \hat{f}(\xi,\psi,\zeta) e^{-ij\xi\frac{\pi}{N}} e^{-ik\psi\frac{\pi}{N}} e^{-il\zeta\frac{\pi}{N}} \\
&= \frac{1}{(2Nh)} \sum_{\xi=-N}^{N} e^{-ij\xi\frac{\pi}{N}} \frac{1}{(2Nh)} \sum_{\psi=-N}^{N} e^{-ik\psi\frac{\pi}{N}} \frac{1}{(2Nh)} \sum_{\zeta=-N}^{N} e^{-il\zeta\frac{\pi}{N}} \hat{f}(\xi,\psi,\zeta) \\
&= \frac{1}{(2Nh)} \sum_{\xi=-N}^{N} e^{-ij\xi\frac{\pi}{N}} \frac{1}{(2Nh)} \sum_{\psi=-N}^{N} e^{-ik\psi\frac{\pi}{N}} \hat{g}(\xi,\psi,l) \\
&= \frac{1}{(2Nh)} \sum_{\xi=-N}^{N} e^{-ij\xi\frac{\pi}{N}} \hat{w}(\xi,k,l) \qquad \text{(A.3)}
\end{aligned}
$$

The function uses a list of equidistant data points **data** separated by a step length $\boldsymbol{h}$. The total number of points is again $m = 2N + 1$ for each index. The following implementation uses the representation of Eq. (A.3).

```
inverseB3DFourier[ data_List, h_, n_] := Block[{ v1, v2, v3},
(* −− iDFT of first coordinate −− *)
v1 = ParallelTable[ Sum[ 1/(2nh) Part[ data, j + n + 1, k + n + 1, l + n + 1]e^(−ilλ π/n),
{l, −n, n}], {j, −n, n}, {k, −n, n}, {λ, −n, n}];
(* −− iDFT of second coordinate −− *)
v2 = ParallelTable[ Sum[ 1/(2nh) Part[ v1, j + n + 1, k + n + 1, λ + n + 1]e^(−ikη π/n),
{k, −n, n}], {j, −n, n}, {η, −n, n}, {λ, −n, n}];
```

```
(* -- iDFT of third coordinate -- *)
v3 = ParallelTable[ Sum[ 1/(2nh) Part[ v2, j + n + 1, η + n + 1, λ + n + 1] e^(-ijξ π/n),
{j, -n, n}], {ξ, -n, n}, {η, -n, n}, {λ, -n, n}];
Chop[ v3]
]
```

Example A.1.10. Discrete Fourier Transform
The example shows how DFT is applied in the N–S algorithm to a set of initial conditions. The next input line uses the discrete data set of Example A.1.8 to demonstrate the use of the function. The input data are the discrete data set **dU03**, the step length $h = 1/2$, and the number of Sinc points $N = 2$.

```
dUo3F = a3DFourier[ dU03, 1/2, 2]
```

{{{0, 0, 0, 0, 0}, {0, 0, 0, 0, 0}, {0, 0, 0, 0, 0}, {0, 0, 0, 0, 0}, {0, 0, 0, 0, 0}},

{{0, 0, 0, 0, 0}, {0, 0, 0, 0, 0}, {0, 1., 0, −1., 0}, {0, 0, 0, 0, 0}, {0, 0, 0, 0, 0}},

{{0, 0, 0, 0, 0}, {0, 1., 0, −1., 0}, {0, 0, 0, 0, 0}, {0, −1., 0, 1., 0}, {0, 0, 0, 0, 0}},

{{0, 0, 0, 0, 0}, {0, 0, 0, 0, 0}, {0, −1., 0, 1., 0}, {0, 0, 0, 0, 0}, {0, 0, 0, 0, 0}},

{{0, 0, 0, 0, 0}, {0, 0, 0, 0, 0}, {0, 0, 0, 0, 0}, {0, 0, 0, 0, 0}, {0, 0, 0, 0, 0}}}

The result is a $5 \times 5 \times 5$ array containing the Fourier amplitudes of the DFT.
The inverse DFT uses the same data structure and thus can be directly applied to the result stored in **dUo3F**

```
idUo3F = inverseB3DFourier[ dUo3F, 1/2, 2]
```

{{{0, 0, 0, 0, 0}, {0, −0.5, 0, 0.5, 0}, {0, 0, 0, 0, 0}, {0, 0.5, 0, −0.5, 0}, {0, 0, 0, 0, 0}},

{{0, −0.5, 0, 0.5, 0}, {0, −1., 0, 1., 0}, {0, −0.5, 0, 0.5, 0}, {0, 0, 0, 0, 0},

{0, −0.5, 0, 0.5, 0}},

{{0, 0, 0, 0, 0}, {0, −0.5, 0, 0.5, 0}, {0, 0, 0, 0, 0}, {0, 0.5, 0, −0.5, 0}, {0, 0, 0, 0, 0}},

{{0, 0.5, 0, −0.5, 0}, {0, 0, 0, 0, 0}, {0, 0.5, 0, −0.5, 0}, {0, 1., 0, −1., 0},

{0, 0.5, 0, −0.5, 0}},

{{0, 0, 0, 0, 0}, {0, −0.5, 0, 0.5, 0}, {0, 0, 0, 0, 0}, {0, 0.5, 0, −0.5, 0}, {0, 0, 0, 0, 0}}}

▽

A.1.7 Spatial Temporal Convolution

The Fourier transform of a 4-d convolution integral results to a product representation in Fourier space. This product allows a direct calculation if the DFT representation and the time representation at Sinc points are known. This simplification of the calculation is used in an implementation of the function **convol4D[]**.

The function **convol4D[]** calculates the convolution product in Fourier space. The function uses the discrete initial conditions given as DFT coefficients and the Green's function given by the relation

$$\hat{v}^{\nu}\left(\bar{\Lambda}, t\right) = \hat{u}^{0,\nu}\left(\bar{\Lambda}\right) \exp\left(-\Lambda^2 \epsilon t\right). \tag{A.4}$$

Here Λ^2 is the square of the matrix elements $\Lambda_{j,k,l}$ generated by the function $\boldsymbol{\Lambda}$**[]**. The time t is replaced by the discrete Sinc points along the time axis generated by $\mathbf{t}\boldsymbol{\mu}$**[]**. The results of $\mathbf{t}\boldsymbol{\mu}$**[]** are stored in the list **tt**. $\boldsymbol{\epsilon}$ is a parameter controlling the convergence on the time line. The total number of Sinc points is given by $\boldsymbol{n}$. The array **Fu** contains the DFTs of the initial conditions $\boldsymbol{u}^0$. The following lines show a parallel implementation of the convolution.

```
convol4D[ λb_, tt_, Fu_, ϵ_, n_] := Block[{},
(* – convolution of time and initial conditions – – *)
Chop[
( ParallelTable [ Table [ Exp [−ϵ( λb[[j, k, l]])^2 tt[[μ]]] Fu[[j, k, l]],
{j, 1, 2n + 1}, {k, 1, 2n + 1}, {l, 1, 2n + 1}], {μ, 1, Length[ tt], 1}])]
]
```

Example A.1.11. 4*D* Convolution in Fourier Space
The example uses the data generated in Examples A.1.9, A.1.5, and A.1.8 as input data. The data are generated by the related *Mathematica* functions defined above. In addition we use a step length $h = 1/2$ and the number of Sinc points $N = 2$. The result of the function is an array of dimension $(2N + 1)^3 \times (2M + 1)$.

```
convol4D[Λ[2, 1/2], tμ[1, 2], dU03, 0.5, 2]
```

```
{{{{−1.96543, −0.987007, 0, 0.987007, 1.96543},
{−1.48051, −0.743489, 0, 0.743489, 1.48051},
{−0.988442, −0.49638, 0, 0.49638, 0.988442},
{−0.493503, −0.24783, 0, 0.24783, 0.493503},
```

{0, 0, 0, 0, 0}}, {{−1.48051, −0.743489, 0, 0.743489, 1.48051},

{−0.991319, −0.99565, 0, 0.99565, 0.991319},

{−0.49638, −0.498549, 0, 0.498549, 0.49638}, {0, 0, 0, 0, 0},

{0.493503, 0.24783, 0, −0.24783, −0.493503}},

{{−0.988442, −0.49638, 0, 0.49638, 0.988442},

{−0.49638, −0.498549, 0, 0.498549, 0.49638}, {0, 0, 0, 0, 0},

{0.49638, 0.498549, 0, −0.498549, −0.49638},

{0.988442, 0.49638, 0, −0.49638, −0.988442}},

{{−0.493503, −0.24783, 0, 0.24783, 0.493503}, {0, 0, 0, 0, 0},

{0.49638, 0.498549, 0, −0.498549, −0.49638},

{0.991319, 0.99565, 0, −0.99565, −0.991319},

{1.48051, 0.743489, 0, −0.743489, −1.48051}}, {{0, 0, 0, 0, 0},

{0.493503, 0.24783, 0, −0.24783, −0.493503},

{0.988442, 0.49638, 0, −0.49638, −0.988442},

{1.48051, 0.743489, 0, −0.743489, −1.48051},

{1.96543, 0.987007, 0, −0.987007, −1.96543}}},

{{{−1.727, −0.89577, 0, 0.89577, 1.727},

{−1.34365, −0.696935, 0, 0.696935, 1.34365},

{−0.906793, −0.470341, 0, 0.470341, 0.906793},

{−0.447885, −0.232312, 0, 0.232312, 0.447885}, {0, 0, 0, 0, 0}},

{{−1.34365, −0.696935, 0, 0.696935, 1.34365},

{−0.929247, −0.963974, 0, 0.963974, 0.929247},

{−0.470341, −0.487918, 0, 0.487918, 0.470341}, {0, 0, 0, 0, 0},

{0.447885, 0.232312, 0, −0.232312, −0.447885}},

{{−0.906793, −0.470341, 0, 0.470341, 0.906793},

{−0.470341, −0.487918, 0, 0.487918, 0.470341}, {0, 0, 0, 0, 0},

{0.470341, 0.487918, 0, −0.487918, −0.470341},

{0.906793, 0.470341, 0, −0.470341, −0.906793}},

{{−0.447885, −0.232312, 0, 0.232312, 0.447885}, {0, 0, 0, 0, 0},

{0.470341, 0.487918, 0, −0.487918, −0.470341},

{0.929247, 0.963974, 0, −0.963974, −0.929247},

{1.34365, 0.696935, 0, −0.696935, −1.34365}}, {{0, 0, 0, 0, 0},

{0.447885, 0.232312, 0, −0.232312, −0.447885},

{0.906793, 0.470341, 0, −0.470341, −0.906793},

{1.34365, 0.696935, 0, −0.696935, −1.34365},

{1.727, 0.89577, 0, −0.89577, −1.727}}},

{{{−0.944733, −0.569783, 0, 0.569783, 0.944733},

{−0.854674, −0.515467, 0, 0.515467, 0.854674},

{−0.606531, −0.365808, 0, 0.365808, 0.606531},

{−0.284891, −0.171822, 0, 0.171822, 0.284891}, {0, 0, 0, 0, 0}},

{{−0.854674, −0.515467, 0, 0.515467, 0.854674},

{−0.687289, −0.829029, 0, 0.829029, 0.687289},

{−0.365808, −0.441248, 0, 0.441248, 0.365808}, {0, 0, 0, 0, 0},

{0.284891, 0.171822, 0, −0.171822, −0.284891}},

{{−0.606531, −0.365808, 0, 0.365808, 0.606531},

{−0.365808, −0.441248, 0, 0.441248, 0.365808}, {0, 0, 0, 0, 0},

{0.365808, 0.441248, 0, −0.441248, −0.365808},

{0.606531, 0.365808, 0, −0.365808, −0.606531}},

{{−0.284891, −0.171822, 0, 0.171822, 0.284891}, {0, 0, 0, 0, 0},

{0.365808, 0.441248, 0, −0.441248, −0.365808},

{0.687289, 0.829029, 0, −0.829029, −0.687289},

{0.854674, 0.515467, 0, −0.515467, −0.854674}}, {{0, 0, 0, 0, 0},

{0.284891, 0.171822, 0, −0.171822, −0.284891},

{0.606531, 0.365808, 0, −0.365808, −0.606531},

{0.854674, 0.515467, 0, −0.515467, −0.854674},

{0.944733, 0.569783, 0, −0.569783, −0.944733}}},

{{{−0.516805, −0.362428, 0, 0.362428, 0.516805},

{−0.543643, −0.38125, 0, 0.38125, 0.543643},

{−0.405693, −0.284507, 0, 0.284507, 0.405693},

{−0.181214, −0.127083, 0, 0.127083, 0.181214}, {0, 0, 0, 0, 0}},

{{−0.543643, −0.38125, 0, 0.38125, 0.543643},

{−0.508333, −0.712975, 0, 0.712975, 0.508333},

{−0.284507, −0.399043, 0, 0.399043, 0.284507}, {0, 0, 0, 0, 0},

{0.181214, 0.127083, 0, −0.127083, −0.181214}},

{{−0.405693, −0.284507, 0, 0.284507, 0.405693},

{−0.284507, −0.399043, 0, 0.399043, 0.284507}, {0, 0, 0, 0, 0},

{0.284507, 0.399043, 0, −0.399043, −0.284507},

{0.405693, 0.284507, 0, −0.284507, −0.405693}},

{{−0.181214, −0.127083, 0, 0.127083, 0.181214}, {0, 0, 0, 0, 0},

{0.284507, 0.399043, 0, −0.399043, −0.284507},

{0.508333, 0.712975, 0, −0.712975, −0.508333},

{0.543643, 0.38125, 0, −0.38125, −0.543643}}, {{0, 0, 0, 0, 0},

{0.181214, 0.127083, 0, −0.127083, −0.181214},

{0.405693, 0.284507, 0, −0.284507, −0.405693},

{0.543643, 0.38125, 0, −0.38125, −0.543643},

{0.516805, 0.362428, 0, −0.362428, −0.516805}}},

{{{−0.45411, −0.328926, 0, 0.328926, 0.45411},

{−0.493389, −0.357377, 0, 0.357377, 0.493389},

{−0.372181, −0.269582, 0, 0.269582, 0.372181},

{−0.164463, −0.119126, 0, 0.119126, 0.164463}, {0, 0, 0, 0, 0}},

{{−0.493389, −0.357377, 0, 0.357377, 0.493389},

{−0.476503, −0.690292, 0, 0.690292, 0.476503},

{−0.269582, −0.390534, 0, 0.390534, 0.269582}, {0, 0, 0, 0, 0},

{0.164463, 0.119126, 0, −0.119126, −0.164463}},

{{−0.372181, −0.269582, 0, 0.269582, 0.372181},

{−0.269582, −0.390534, 0, 0.390534, 0.269582}, {0, 0, 0, 0, 0},

{0.269582, 0.390534, 0, −0.390534, −0.269582},

{0.372181, 0.269582, 0, −0.269582, −0.372181}},

{{−0.164463, −0.119126, 0, 0.119126, 0.164463}, {0, 0, 0, 0, 0},

{0.269582, 0.390534, 0, −0.390534, −0.269582},

{0.476503, 0.690292, 0, −0.690292, −0.476503},

{0.493389, 0.357377, 0, −0.357377, −0.493389}}, {{0, 0, 0, 0, 0},

{0.164463, 0.119126, 0, −0.119126, −0.164463},

{0.372181, 0.269582, 0, −0.269582, −0.372181},

{0.493389, 0.357377, 0, −0.357377, −0.493389},

{0.45411, 0.328926, 0, −0.328926, −0.45411}}}}

▽

A.1.8 Sinc Functions

This section collects several basic function definitions useful in interpolation and integration applications. The basic function we use in all kind of Sinc approximations is the scaled Sinc function $\mathrm{Sinc}(x)$. *Mathematica* offers a predefined **Sinc[]** function defined as $\sin(x)/x$. However, Sinc approximations are based on a π scaled definition. This means we have to redefine the Sinc function as a scaled version according to the definition

$$\mathrm{Sinc}(x) = \begin{cases} \frac{\sin(\pi x)}{\pi x} & \text{if } x \neq 0 \\ 1 & \text{if } x = 0. \end{cases} \tag{A.5}$$

The next line defines the scaled Sinc function, we call it **sSinc[].** The scaled version is defined in *Mathematica* as

sSinc[x_] := Sinc[πx]

The behavior is as expected

sSinc[*x*]

Sinc[πx]

and the limit approaching to zero

Limit[sSinc[*x*], $x \to$ 0]

1

Potential applications of the scaled Sinc function are all kind of integrations. An indefinite integral is for example useful to approximate integral equations. Integrals of special interests here are also convolution integrals. To represent indefinite integrals we use the integral of the Sinc function in the following way [5]

$$\sigma(k) = \int_0^k \frac{\sin(\pi x)}{\pi x}\, dx = \int_0^k \mathrm{Sinc}(x)\, dx = \frac{1}{\pi}\,\mathrm{Si}(\pi k) \tag{A.6}$$

which results to the Sine integral evaluated at πk. Since *Mathematica* offers the Sine integral as a pre-defined function we will make use of this definition to save time in integrations. The function **σ[]** uses an upper limit $k \in \mathbb{Z}$. The related *Mathematica* definition follows directly from (A.6) as

σ[k_] := SetPrecision[SinIntegral[πk]/π, 120]

We used a high precision representation to keep track with the numerical needs. Based on **σ[]** a function **e[]** is used in the definition of indefinite integrals [5]. The function **e[]** is based on **σ[]** by simply adding $1/2$ to the Sine integral. This basic definition is given by

e[k_Integer] :=$\frac{1}{2}$ + σ[*k*]

Next the function **e[]** is used to generate a Töplitz matrix **mT[]**. Töplitz matrices are the basis in Sinc methods to represent indefinite integrals. The function **mT[]** uses two integers $i_{\max}$ and $j_{\max}$ specifying the maximal dimension of the resulting matrix $m \times n$ with $m = 2i_{\max} + 1$ and $n = 2j_{\max} + 1$. The function **mT[]** is based on **e[]** and **σ[]**.

The following definition uses a precision of 120 Digit

mT[imax_, jmax_] :=*N*[Table[*e*[*i* − *j*], {*i*, − imax, imax}, {*j*, − jmax, jmax}], 120]

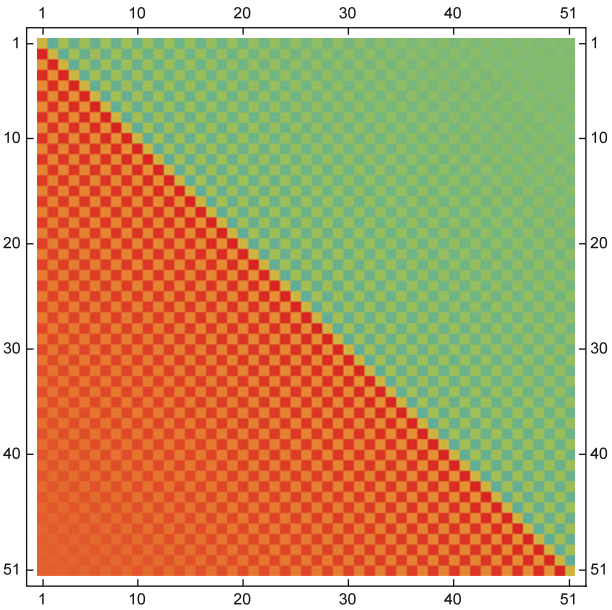

Fig. A.2 Töplitz matrix

To depict the symmetry properties of Töplitz matrices we use a graphical representation of a $m \times m$ matrix with $m = 2N + 1$ and $N = 25$ (Fig. A.2). The next line generates a matrix plot of the Töplitz matrix **mT[]**

MatrixPlot[mT[25, 25], ColorFunction → "Rainbow"]

It is apparent that Töplitz matrices are by the diagonal divided into an upper and a lower triangular part. The diagonal elements have all the same value as well as the off diagonal tracks parallel to the diagonal. This becomes obvious if we truncate the valid digits from 120 to 3 and form a matrix representation of a smaller dimension.

***N*[mT[2, 2], 3] // MatrixForm**

$$\begin{pmatrix} 0.500 & -0.0895 & 0.0486 & -0.0331 & 0.0250 \\ 1.09 & 0.500 & -0.0895 & 0.0486 & -0.0331 \\ 0.951 & 1.09 & 0.500 & -0.0895 & 0.0486 \\ 1.03 & 0.951 & 1.09 & 0.500 & -0.0895 \\ 0.975 & 1.03 & 0.951 & 1.09 & 0.500 \end{pmatrix}$$

A.1.9 Conformal Maps

Conformal maps in Sinc methods are used to map discrete points from the real line to a domain of interest. The representation of discrete points on an interval are generated by mapping the discrete real line to such an interval. These maps allow us to use the whole range in $\mathbb{R}$ to be mapped for example onto a finite interval. Such kind of mappings are extremely useful in practical applications. The advantage of these mappings is that the convergence is improved and becomes nearly optimal [8]. The conformal map ϕ is invertible with $\psi = \phi^{-1}$. So that an equidistant distribution of points on $\mathbb{R}$ is mapped to a non-equidistant distribution of points on a finite interval. Since we are interested to apply this concept not only for finite intervals but also for semi-infinite, or infinite intervals we introduce the following definitions [5, 7].

The next *Mathematica* function **ϕ[]** defines a set of standard conformal maps for different domain types. The domains are specified by the endpoints a and b and are defined for a finite interval $x \in (a, b)$ as

$$\phi_{a,b}(x) = \ln\left(\frac{x-a}{b-x}\right) \quad \text{for } x \in (a, b) \tag{A.7}$$

for a semi-infinite interval $x \in (a, \infty)$ with $a \in \mathbb{R}_+$ the conformal map is

$$\phi_{a,\infty}(x) = \begin{cases} \ln(x-a) & \text{for an algebraic decay at } \infty \\ \ln(\sinh(x-a)) & \text{for an exponential decay at } \infty \end{cases} \quad \text{and } x \in (a, \infty), \tag{A.8}$$

and for an infinite interval $x \in (-\infty, \infty)$ we have

$$\phi_{-\infty,\infty}(x) = \begin{cases} \ln\left(\sqrt{x^2+1}+x\right) & \text{for an algebraic decay at } \pm\infty \\ x & \text{for an exponential decay at } \pm\infty \end{cases} \quad \text{with } x \in (-\infty, \infty). \tag{A.9}$$

The conformal map **ϕ[]** defined below uses as input three arguments and one option. The variable x and the two boundary points a and b are mandatory input variables. The *Mathematica* definition of the conformal map combines the definitions given in (A.7)–(A.9) in a piecewise defined function. The *Mathematica* function thus is able to decide which type of mapping should be used. This decision is based on the boundary values of the interval.

```
Options[ϕ] = { AlgebraicDecay → False};

ϕ[ x_, a_, b_, OptionsPattern[]] :=

Block[{ algebraicDecay}, algebraicDecay = OptionValue[ AlgebraicDecay];
```

$$\textbf{Piecewise}\left[\begin{pmatrix} \log\left(\frac{x-a}{b-x}\right) & -\infty < a \wedge b < \infty \\ \textbf{If}\left[\textbf{algebraicDecay}, \log(x-a), \log(\sinh(x-a))\right] & (0 = a \wedge b = \infty) \vee (a > 0 \wedge a < \infty \wedge b = \infty) \\ \textbf{If}\left[\textbf{algebraicDecay}, \log\left(\sqrt{x^2+1}+x\right), x\right] & -\infty = a \wedge b = \infty \end{pmatrix}\right]$$

The conformal map is useful for symbolic as well as numeric calculations. **ϕ[]** delivers with its default settings the following functional relations depending on the interval limits

$$\{\phi[z, 0, 1], \phi[z, 0, \infty], \phi[z, -\infty, \infty]\}$$

$$\left\{\text{Log}\left[\frac{z}{1-z}\right],\ \text{Log}[\text{Sinh}[z]],\ z\right\}$$

If we set the option **AlgebraicDecay** to **True**, the same input line in *Mathematica* delivers the result for an algebraic decay at infinity.

$$\textbf{SetOptions}[\phi,\ \textbf{AlgebraicDecay} \rightarrow\ \textbf{True}]$$

$$\{\ \text{AlgebraicDecay} \rightarrow\ \text{True}\}$$

Then the generated functions for the semi-infinite and infinite interval changes to

$$\{\phi[z, 0, 1], \phi[z, 0, \infty], \phi[z, -\infty, \infty]\}$$

$$\left\{\text{Log}\left[\frac{z}{1-z}\right],\ \text{Log}[z],\ \text{Log}\left[z+\sqrt{1+z^2}\right]\right\}$$

In a numerical application the conformal map **ϕ[]** delivers depending again on the interval limits numerical values for these functions. The following line demonstrates this for a finite interval with rational and real-valued numbers

$$\left\{\phi\left[\tfrac{1}{2}, -1, 1\right], \phi[0.3, -1, 1]\right\}$$

$$\{\ \text{Log}[3], 0.619039\}$$

The inverse conformal map $\psi_{a,b}(x) = \phi^{-1}_{a,b}(x)$ is also defined as a piecewise function for the different interval types. The corresponding map is selected according to the specified boundary points a and b. In the definition given below three standard domains are implemented: a finite, a semi-infinite, and an infinite domain with algebraic or non-algebraic decay. The definitions are as follows

$$\psi_{a,b}(x) = \frac{a + be^x}{1 + e^x} \quad \text{for } x \in (a, b) \tag{A.10}$$

for a semi-infinite interval $x \in (a, \infty)$ with $a \in \mathbb{R}_+$ the inverse conformal map is

$$\psi_{a,\infty}(x) = \begin{cases} a + e^x & \text{for an algebraic decay at } \infty \\ \log\left(\sqrt{e^{2x}+1} + e^x + a\right) & \text{for a non-algebraic decay at } \infty \end{cases} \quad \text{for } x \in (a, \infty), \tag{A.11}$$

and for an infinite interval $x \in (-\infty, \infty)$ we have

$$\psi_{-\infty,\infty}(x) = \begin{cases} \sinh(x) & \text{for an algebraic decay at } \pm\infty \\ x & \text{for an exponential decay at } \pm\infty \end{cases} \quad \text{for } x \in (-\infty, \infty). \tag{A.12}$$

The function **ψ[]** uses three arguments which are mandatory: the variable x and the two boundary points a and b. A fourth optional argument can be used to select the algebraic decay for large arguments. Again we define **ψ[]** as a piecewise function in the following lines.

```
Options[ψ] = { AlgebraicDecay → False};

ψ[ x_, a_, b_, OptionsPattern[]] :=

Block[{ algebraicDecay}, algebraicDecay = OptionValue[ AlgebraicDecay];

Piecewise[
(
{ (a+b e^x)/(e^x+1),                                   -∞ < a ∧ b < ∞
{ If [algebraicDecay, a + e^x, log(√(e^(2x)+1) + e^x + a)],   (0 = a ∧ b = ∞) ∨ (a > 0 ∧ a < ∞ ∧ b = ∞)
{ If [algebraicDecay, sinh(x), x],                     -∞ = a ∧ b = ∞
)
]]
```

The inverse conformal map is useful for symbolic as well as for numeric calculations. It delivers with its default settings the following functional relations depending on the interval limits

```
{ψ[w, 0, 1], ψ[w, 0, ∞], ψ[w, -∞, ∞]}
```

$$\left\{\frac{e^w}{1+e^w},\ \mathrm{Log}\left[e^w + \sqrt{1+e^{2w}}\right], w\right\}$$

If we set the option **AlgebraicDecay** to **True**

```
SetOptions[ψ, AlgebraicDecay → True]
```

{ AlgebraicDecay → True}

Then the generated functions for the semi-infinite and infinite interval change.

$$\{\psi[w,0,1],\psi[w,0,\infty],\psi[w,-\infty,\infty]\}$$

$$\left\{\frac{e^w}{1+e^w},\ e^w,\ \text{Sinh}[w]\right\}$$

In a numerical application the inverse conformal map function $\psi[]$ delivers according to the interval limits numerical values for these functions. The inverse conformal map specifically is used to generate the Sinc points in numerical calculations. The following example demonstrates the use for a finite interval with rational and real-valued numbers

$$\left\{\psi\left[\tfrac{1}{2},-1,1\right],\psi[0.3,-1,1]\right\}$$

$$\left\{\frac{-1+\sqrt{e}}{1+\sqrt{e}},0.148885\right\}$$

A sequence of Sinc points $m = 2N + 1$ with $N = 3$ on the interval $z \in [-1, 1]$ is generated as follows

$$\mathbf{zk = Table}[\psi[kh,-1,1],\{k,-3,3\}]$$

$$\left\{\frac{-1+e^{-3h}}{1+e^{-3h}},\ \frac{-1+e^{-2h}}{1+e^{-2h}},\ \frac{-1+e^{-h}}{1+e^{-h}},\ 0,\ \frac{-1+e^{h}}{1+e^{h}},\ \frac{-1+e^{2h}}{1+e^{2h}},\ \frac{-1+e^{3h}}{1+e^{3h}}\right\}$$

The result is a sequence of symbolic expressions containing h, the step length, as an arbitrary parameter. If we specify the step length by $h = \pi \Big/ \sqrt{N}$ we will get for the same setup

$$\mathbf{zk = Table}\left[\psi\left[k\pi\Big/\sqrt{3},-1,1\right],\{k,-3,3\}\right]$$

$$\left\{\frac{-1+e^{-\sqrt{3}\pi}}{1+e^{-\sqrt{3}\pi}},\ \frac{-1+e^{-\frac{2\pi}{\sqrt{3}}}}{1+e^{-\frac{2\pi}{\sqrt{3}}}},\ \frac{-1+e^{-\frac{\pi}{\sqrt{3}}}}{1+e^{-\frac{\pi}{\sqrt{3}}}},\ 0,\ \frac{-1+e^{\frac{\pi}{\sqrt{3}}}}{1+e^{\frac{\pi}{\sqrt{3}}}},\ \frac{-1+e^{\frac{2\pi}{\sqrt{3}}}}{1+e^{\frac{2\pi}{\sqrt{3}}}},\ \frac{-1+e^{\sqrt{3}\pi}}{1+e^{\sqrt{3}\pi}}\right\}$$

or in floating point representation

$$\mathbf{N[\,zk]}$$

$$\{-0.991371,-0.948217,-0.719641,0.,0.719641,0.948217,0.991371\}$$

The following helper function $\rho[]$ is the exponential of a conformal map. The function is used in the representation of the basis in Sinc approximations. The function uses three arguments with the same meaning as for a conformal map.

$$\boldsymbol{\rho(\ x_,\ a_,\ b_)} := e^{\phi(x,a,b)}$$

The next lines show the application of conformal maps. Let us start by selecting the map ϕ for a finite interval $[a, b] = [0, \infty]$. The map function $\phi[]$ is then used as

$$\boldsymbol{\phi(x,0,\infty)}$$

$\log(x)$

It delivers the analytic expression for the conformal map. The inverse conformal map for the same interval works for symbolic discrete values $x_k = kh$ as follows

$\boldsymbol{\psi(hk, 0, 1)}$

$\frac{e^{hk}}{e^{hk}+1}$

The generated function is a combination of exponential functions using the discrete values k and the step length h in the representation. The exponential version of the conformal map $\rho_{a,b}(x)$ delivers the exponential representation of ϕ as follows

$\boldsymbol{\rho(x, -\infty, \infty)}$

$x + \sqrt{x^2 + 1}$

A.1.10 Basis Functions

To represent an approximation of a function $f(x)$ by shifted Sinc functions, we define a set of basis functions [5]. This set of shifted Sinc functions establishes the basis of a Sinc approximation. The following *Mathematica* function $\boldsymbol{\gamma[]}$ is the generator of the basis. The symbolic expression behind the generator is

$$\gamma_{a,b}(j,h) \circ \phi_{a,b}(x) = \text{Sinc}\left((\phi_{a,b}(x) - jh)/h\right) = S(j,h) \circ \phi_{a,b}(x). \tag{A.13}$$

The basis function γ uses five input arguments: the independent variable x, the shifting index j, the step length h, and the boundary points a and b. The *Mathematica* definition $\boldsymbol{\gamma[]}$ uses these dependencies as input data. The next line defines the basis $\boldsymbol{\gamma[]}$

$\boldsymbol{\gamma[\,x_,\ j_,\ h_,\ a_,\ b_] := \text{sSinc}[(\phi[x, a, b] - jh)/h]}$

We generate the set of basis functions by a variation of the shifting index j in the range $-N \le j \le N$. A useful *Mathematica* function to generate the set is the function **Table[]**.

Example A.1.12. Sinc Basis Set
As a specific example we use $N = 1$ and keep the boundaries a and b arbitrary

$\boldsymbol{\text{Table}[\gamma(x, j, h, a, b), \{j, -1, 1\}]}$

$$\left\{ \operatorname{sinc}\left(\frac{\pi}{h}\left(\begin{cases} \log\left(\frac{x-a}{b-x}\right) & -\infty < a \wedge b < \infty \\ \log(x-a) & (0 = a \wedge b = \infty) \vee \\ & (a > 0 \wedge a < \infty \wedge b = \infty) \\ \log\left(x + \sqrt{x^2+1}\right) & -\infty = a \wedge b = \infty \end{cases} + h \right) \right) \right.,$$

$$\operatorname{sinc}\left(\frac{\pi}{h}\left(\begin{cases} \log\left(\frac{x-a}{b-x}\right) & -\infty < a \wedge b < \infty \\ \log(x-a) & (0 = a \wedge b = \infty) \vee \\ & (a > 0 \wedge a < \infty \wedge b = \infty) \\ \log\left(x + \sqrt{x^2+1}\right) & -\infty = a \wedge b = \infty \end{cases} \right) \right),$$

$$\operatorname{sinc}\left(\frac{\pi}{h}\left(\begin{cases} \log\left(\frac{x-a}{b-x}\right) & -\infty < a \wedge b < \infty \\ \log(x-a) & (0 = a \wedge b = \infty) \vee \\ & (a > 0 \wedge a < \infty \wedge b = \infty) \\ \log\left(x + \sqrt{x^2+1}\right) & -\infty = a \wedge b = \infty \end{cases} - h \right) \right)$$

Since the interval $[a, b]$ is not specified the conformal map is inserted in the basis functions as a piecewise defined function. This behavior allows us to set up an approximation independent of the specific interval.
▽

In some cases the shifted Sinc basis is altered at the end points to adapt the boundary conditions in a better way. The next function we introduce represents a piecewise representation of the basis function for Sinc approximation. The function is defined for the end points at M and N in a special way and uses for the internal points the definition introduced by $\gamma_{a,b}(j,h) \circ (x)$ in Eq. (A.13). Specifically such an altered basis is given by

$$\omega_{a,b}^{M,N}(j,h)\circ(x) = \begin{cases} \frac{1}{1+\rho(z)} - \sum_{k=-M+1}^{N} \frac{1}{1+e^{kh}} \gamma_{a,b}(k,h) \circ (x) & \text{if } j = -M \\ \gamma_{a,b}(j,h) \circ (x) & \text{if } j = -M+1, \ldots, N-1 \\ \frac{\rho(z)}{1+\rho(z)} - \sum_{k=-M}^{N-1} \frac{e^{kh}}{1+e^{kh}} \gamma_{a,b}(j,h) \circ (x) & \text{if } j = N. \end{cases} \tag{A.14}$$

In addition to γ the function **omega[]** uses special expressions for the first and last Sinc point of the approximation to accommodate the boundary properties in a more effective way. This means ω uses a piecewise defined function for the smallest and largest integer number M and N, and the bulk points in between. The *Mathematica*

definition is given next **omega(x_, k_, h_, a_, b_, Mx_, Nx_) :=**

$$\textbf{Piecewise}\left[\left(\begin{array}{ll} \gamma(x,k,h,a,b) & 1-\mathrm{Mx}\le k\le \mathrm{Nx}-1 \\ \frac{1}{\rho(x,a,b)+1}-\sum_{j=1-\mathrm{Mx}}^{\mathrm{Nx}}\frac{\gamma(x,j,h,a,b)}{e^{hj}+1} & k=-\mathrm{Mx} \\ \frac{\rho(x,a,b)}{\rho(x,a,b)+1}-\sum_{j=-\mathrm{Mx}}^{\mathrm{Nx}-1}\frac{e^{hj}\gamma(x,j,h,a,b)}{e^{hj}+1} & k=\mathrm{Nx} \end{array}\right)\right]$$

The next example demonstrates the application of the adapted Sinc basis to different types of domains.

Example A.1.13. Adapted Sinc Basis Set
A typical application of **omega[]** to the input data with $N = M = 2$, an interval $x \in [-1, 1]$, but still a symbolic step length h, delivers the basis set. The basis for such a set of input data is then generated using *Mathematic*'s function **Table[]** as

basis = Simplify[Table[omega(x, k, h, −1, 1, 2, 2), {k, −2, 2}]]

$$\left\{\frac{1}{2}\left(-\operatorname{sinc}\left(\frac{\pi\log\left(\frac{x+1}{1-x}\right)}{h}\right)-\frac{2\operatorname{sinc}\left(\frac{\pi\left(\log\left(\frac{x+1}{1-x}\right)-h\right)}{h}\right)}{e^h+1}\right.\right.$$
$$\left.-\frac{2e^h\operatorname{sinc}\left(\frac{\pi\left(h+\log\left(\frac{x+1}{1-x}\right)\right)}{h}\right)}{e^h+1}-\frac{2\operatorname{sinc}\left(\frac{\pi\left(\log\left(\frac{x+1}{1-x}\right)-2h\right)}{h}\right)}{e^{2h}+1}-x+1\right),$$
$$\operatorname{sinc}\left(\frac{\pi\left(h+\log\left(\frac{x+1}{1-x}\right)\right)}{h}\right),\ \operatorname{sinc}\left(\frac{\pi\log\left(\frac{x+1}{1-x}\right)}{h}\right),\ \operatorname{sinc}\left(\frac{\pi\left(\log\left(\frac{x+1}{1-x}\right)-h\right)}{h}\right),$$
$$\frac{1}{2}\left(-\operatorname{sinc}\left(\frac{\pi\log\left(\frac{x+1}{1-x}\right)}{h}\right)-\frac{2e^h\operatorname{sinc}\left(\frac{\pi\left(\log\left(\frac{x+1}{1-x}\right)-h\right)}{h}\right)}{e^h+1}-\frac{2\operatorname{sinc}\left(\frac{\pi\left(h+\log\left(\frac{x+1}{1-x}\right)\right)}{h}\right)}{e^h+1}\right.$$
$$\left.\left.-\frac{2\operatorname{sinc}\left(\frac{\pi\left(2h+\log\left(\frac{x+1}{1-x}\right)\right)}{h}\right)}{e^{2h}+1}+x+1\right)\right\}$$

The same basis on a semi-infinite interval $x \in [0, \infty)$ is

basis = Simplify[Table[omega(x, k, h, 0, ∞, 2, 2), {k, −2, 2}]]

$$\left\{-\frac{1}{2}\operatorname{sinc}\left(\frac{\pi\log(x)}{h}\right)-\frac{\operatorname{sinc}\left(\frac{\pi(\log(x)-h)}{h}\right)}{e^h+1}-\frac{e^h\operatorname{sinc}\left(\frac{\pi(h+\log(x))}{h}\right)}{e^h+1}-\frac{\operatorname{sinc}\left(\frac{\pi(\log(x)-2h)}{h}\right)}{e^{2h}+1}+\frac{1}{x+1},\right.$$
$$\operatorname{sinc}\left(\frac{\pi(h+\log(x))}{h}\right),\ \operatorname{sinc}\left(\frac{\pi\log(x)}{h}\right),\ \operatorname{sinc}\left(\frac{\pi(\log(x)-h)}{h}\right),$$
$$\left.-\frac{1}{2}\operatorname{sinc}\left(\frac{\pi\log(x)}{h}\right)-\frac{e^h\operatorname{sinc}\left(\frac{\pi(\log(x)-h)}{h}\right)}{e^h+1}-\frac{\operatorname{sinc}\left(\frac{\pi(h+\log(x))}{h}\right)}{e^h+1}-\frac{\operatorname{sinc}\left(\frac{\pi(2h+\log(x))}{h}\right)}{e^{2h}+1}+\frac{x}{x+1}\right\}$$

and if we have a non-algebraic decay we set

SetOptions[ϕ, AlgebraicDecay → False]

{ AlgebraicDecay → False}

The corresponding basis becomes

basis = Simplify[Table[omega($x, k, h, 0, \infty, 2, 2$), {$k, -2, 2$}]]

$$\left\{ -\frac{1}{2}\ \mathrm{sinc}\left(\frac{\pi \log(\sinh(x))}{h}\right) - \frac{\mathrm{sinc}\left(\frac{\pi(\log(\sinh(x))-h)}{h}\right)}{e^h+1} - \frac{e^h\ \mathrm{sinc}\left(\frac{\pi(h+\log(\sinh(x)))}{h}\right)}{e^h+1} \right.$$
$$- \frac{\mathrm{sinc}\left(\frac{\pi(\log(\sinh(x))-2h)}{h}\right)}{e^{2h}+1} + \frac{1}{\sinh(x)+1},$$
$$\mathrm{sinc}\left(\frac{\pi(h+\log(\sinh(x)))}{h}\right),\ \mathrm{sinc}\left(\frac{\pi \log(\sinh(x))}{h}\right),\ \mathrm{sinc}\left(\frac{\pi(\log(\sinh(x))-h)}{h}\right),$$
$$-\frac{1}{2}\ \mathrm{sinc}\left(\frac{\pi \log(\sinh(x))}{h}\right) - \frac{e^h\ \mathrm{sinc}\left(\frac{\pi(\log(\sinh(x))-h)}{h}\right)}{e^h+1} - \frac{\mathrm{sinc}\left(\frac{\pi(h+\log(\sinh(x)))}{h}\right)}{e^h+1}$$
$$\left. - \frac{\mathrm{sinc}\left(\frac{\pi(2h+\log(\sinh(x)))}{h}\right)}{e^{2h}+1} + \frac{\sinh(x)}{\sinh(x)+1} \right\}$$

▽

The following example demonstrates in short how a general Sinc approximation is setup.

Example A.1.14. Sinc Approximation
A general Sinc approximation uses a dot product of the expansion coefficients $f_k = f(z_k)$ with the basis set. For this example we used a non-algebraic decay of the conformal map at ∞ which is equal to the basis generated in Example A.1.13

Table [$f\,(z_k)$, {$k, -2, 2$}] . basis

$$f\,(z_0)\ \mathrm{sinc}\left(\frac{\pi \log(\sinh(x))}{h}\right) + f\,(z_1)\ \mathrm{sinc}\left(\frac{\pi(\log(\sinh(x))-h)}{h}\right)$$
$$+ f\,(z_{-1})\ \mathrm{sinc}\left(\frac{\pi(h+\log(\sinh(x)))}{h}\right)$$
$$+ f\,(z_{-2}) \left(-\frac{1}{2}\ \mathrm{sinc}\left(\frac{\pi \log(\sinh(x))}{h}\right) - \frac{\mathrm{sinc}\left(\frac{\pi(\log(\sinh(x))-h)}{h}\right)}{e^h+1} - \frac{e^h\ \mathrm{sinc}\left(\frac{\pi(h+\log(\sinh(x)))}{h}\right)}{e^h+1} \right.$$
$$\left. - \frac{\mathrm{sinc}\left(\frac{\pi(\log(\sinh(x))-2h)}{h}\right)}{e^{2h}+1} + \frac{1}{\sinh(x)+1} \right)$$
$$+ f\,(z_2) \left(-\frac{1}{2}\ \mathrm{sinc}\left(\frac{\pi \log(\sinh(x))}{h}\right) - \frac{e^h\ \mathrm{sinc}\left(\frac{\pi(\log(\sinh(x))-h)}{h}\right)}{e^h+1} - \frac{\mathrm{sinc}\left(\frac{\pi(h+\log(\sinh(x)))}{h}\right)}{e^h+1} \right.$$
$$\left. - \frac{\mathrm{sinc}\left(\frac{\pi(2h+\log(\sinh(x)))}{h}\right)}{e^{2h}+1} + \frac{\sinh(x)}{\sinh(x)+1} \right)$$

where $f(z_k)$ with $-N \leq k \leq N$ are the unknown expansion coefficients of a function f. Such kind of representations can be directly used in symbolic calculations in connections with approximations.
$\triangledown$

A.1.11 Laplace Transform

Laplace's transform is essential in the calculation of Sinc convolution integrals. The use of this transform in connection with one-dimensional approximations goes back to Lubich [3, 4]. Here, we will use a 3-d version of the Laplace transform in a slightly different way [5]. The main application of Laplac's transform is the discrete representation of the convolution kernel. If the symbolic representation of a Laplace transform is known the discrete representation uses the eigenvalues of the Töplitz matrices introduced above. If a symbolic Laplace transform of the kernel is not known a numeric representation via Sinc approximations can be used [1, 2]. A three-dimensional Laplace transform of a function f for our applications is defined as follows

$$\mathscr{L}f(x,y,z) = \mathscr{F}(s,t,u) = \int_o^{c_x}\int_0^{c_y}\int_0^{c_z} f(x,y,z)e^{-\frac{x}{s}}e^{-\frac{y}{t}}e^{-\frac{z}{u}}dzdydx. \tag{A.15}$$

The discrete version of the Laplace transform uses the representation [5]

$$\mathscr{F}(s,t,u) \approx \mathscr{F}\left(s_i,t_j,u_k\right) \quad \text{with} \quad -N \leq i,j,k \leq N \tag{A.16}$$

where s_i, t_j, and u_k are the eigenvalues of $hI^{(-1)}.D(1/\phi') = A_m$, so that $A_m = X.D(s_i).X^{-1}$ is a diagonal representation of the eigenvector matrix and the corresponding eigenvalues. If $\mathscr{F}$ is not explicitly known, then the values $\mathscr{F}(s_i)$ for each dimension can be approximated to arbitrary accuracy via use of Sinc quadrature applied to the "Laplace transform" integral (see [5, Sect. 1.5.7]).

The implementation of the 3-d Laplace transform uses the outlines given in Chap. 5. The following function generates a discrete representation of this Laplace transform based on the eigenvalues of the used discrete approximation. The function **laplaceTransform3D**[] uses a symbolic representation assigned to the variable **ltIn** representing the integrand of the Laplace integral. The continuous Laplace variables are replaced here by their discrete eigenvalues in each direction. Since the Laplace variables are independent from each other such kind of operation can be performed in parallel.

```
laplaceTransform3D[ ltIn_, eigenValuesX_, eigenValuesY_, eigenValuesZ_] :=

Block[{t, s, u, lT, mX, precissionGoal = 120},

lT = ltIn;
```

```
(* -- replacement of Laplace variables by Eigenvalues -- *)
mX =
SetPrecision[
ParallelTable[
lT /.{s → eigenValuesX[[j]], t → eigenValuesY[[k]], u → eigenValuesZ[[l]]},
{j, 1, Length[ eigenValuesX]}, {k, 1, Length[ eigenValuesY]},
{l, 1, Length[ eigenValuesZ]}], precissionGoal];

mX
]
```

For the function defined above it is important to know the explicit representation of the Laplace transform of the kernel. In our application the kernel is given by the free space Green's function $G = 1/(4\pi r)$ with $r = \sqrt{x^2 + y^2 + z^2}$. The following section introduces the explicit representation of the Laplace transform for this kernel.

A.1.12 Free Space Green's Function for Poisson's Equation

The next three *Mathematica* functions implement the Laplace transform of the Green's function $G = 1/(4\pi r)$ on $\mathbb{R}^3$. The functions are set up using the integration algorithm given in [7].

The first function **Qfunc[]** is used as one of the functions to generate the Laplace variables. The function **Qfunc[]** is a helper function implementing the properties of Theorem 2.4.1 of [7]. The function uses a single in general complex argument to distinguish between the different domain properties of its argument. The main point of Theorem 2.4.1 is that the complex plane is partitioned into different sub-domains. These regions are each assigned to an analytic expression as follows

```
Qfunc( a_) := Block[{ rEa, iMa, La}, rEa = ℜ(a);
iMa = ℑ(a);
```

$$\mathbf{La} = \mathbf{iMa} + \mathbf{rEa} - 1;$$

$$\mathbf{Piecewise}\left[\left[\begin{pmatrix} \infty & a = 1 \vee a = i \\ \log\left(\left|\frac{i-a}{1-a}\right|\right) + i\left(2\pi - \arg\left(\frac{-a+i}{1-a}\right)\right) & |a| < 1 \wedge \mathbf{La} > 0 \\ \log\left(\left|\frac{i-a}{1-a}\right|\right) + i\pi & \mathbf{La} = 0 \wedge |a| < 1 \\ \log\left(\left|\frac{i-a}{1-a}\right|\right) + \frac{i\pi}{2} & \mathbf{La} > 0 \wedge |a| = 1 \\ \log\left(\frac{-a+i}{1-a}\right) & \mathbf{True} \end{pmatrix}\right]\right];$$

The function **lapTrInvR2[]** is a second helper function linked to the function **Qfunc[]**. The function uses two arguments and generates the main terms of the Laplace transformed Green's function. The definition is implementing the more effective version of the Laplace transform where the standard Laplace variables are replaced by its reciprocals.

$$\mathbf{lapTrInvR2[\,u_,\ v_]} := \mathbf{Block[\{\,lam,\ zeta,\ G1\},}$$

$$\mathbf{lam} = \sqrt{\left(\frac{1}{u^2} + \frac{1}{v^2}\right)};$$

$$\mathbf{zeta} = \sqrt{\frac{\left(\frac{1}{u} + \frac{i}{v}\right)}{\frac{1}{u} - \frac{i}{v}}};$$

$$\mathbf{G1} = \frac{-1}{\mathbf{lam}}\left(\mathbf{Qfunc}[i * \mathbf{zeta}] - \mathbf{Qfunc}[-i * \mathbf{zeta}]\right)\Big];$$

The actual setup of the Laplace transform for the free space Green's function of the Poisson equation is compiled in the following function **lapTr3Poi[]**. The function uses the three Laplace variables in standard notation. The Laplace representation of Green's function is generated by using the functions **lapTrInvR2[]** and **Qfunc[]** to set up the definition consisting of three terms plus a pre-factor.

$$\mathbf{lapTr3Poi[\,u_,\ v_,\ w_]} := \mathbf{Block[\{\,H1,\ H2,\ H3, \mathit{R},\ G1\},}$$

$$\mathbf{H1} = \mathbf{lapTrInvR2}[v, w]/u;$$

$$\mathbf{H2} = \mathbf{lapTrInvR2}[w, u]/v;$$

$$\mathbf{H3} = \mathbf{lapTrInvR2}[u, v]/w;$$

$$R = \frac{1}{u^2} + \frac{1}{v^2} + \frac{1}{w^2};$$

$$\mathbf{G1} = \frac{1}{R}\left(\frac{-1}{8} + \frac{(\mathbf{H1} + \mathbf{H2} + \mathbf{H3})}{4\pi}\right)\Big];$$

The application of the function to the three Laplace variables delivers a general analytic expression for the Laplace transform of the free space Green's function.

lapTr3Poi(s, t, u);

We suppress the huge symbolic output here. If we are interested in numerical values as in convolution integral representations the function delivers a complex number

lapTr3Poi[.1, .2, 1 + i]

$0.00362125 + 0.00109769i$

A.1.13 Convolution Integrals

The *Mathematica* function discussed in this section implements a 3-d convolution integral of the following form

$$p(x,y,z) = \int_a^b \int_c^d \int_e^f G(|\boldsymbol{r} - \boldsymbol{r}'|) g(x',y',z') dx' dy' dz' = I. \tag{A.17}$$

where G represents the Green's function and g for example is the right-hand side of the Poisson equation $\Delta p = -g$. The intervals $[a,b]$, $[c,d]$, and $[e,f]$ may be finite or infinite. The algorithm for the one-dimensional case is described in [6]. The higher dimensional algorithm is discussed in [7].The following function uses this algorithms for the 3-d case. The idea basically is the representation of the 3-d convolution integral by eight different variations of indefinite convolution integrals. Symbolically this can be written as

$$\begin{aligned}
I = &\int_a^x \int_c^y \int_e^z g(\xi,\eta,\kappa) G(x,y,z,\xi,\eta,\kappa) d\eta d\xi d\kappa \\
&+ \int_x^b \int_c^y \int_e^z g(\xi,\eta,\kappa) G(x,y,z,\xi,\eta,\kappa) d\eta d\xi d\kappa \\
&+ \int_x^b \int_y^d \int_e^z g(\xi,\eta,\kappa) G(x,y,z,\xi,\eta,\kappa) d\eta d\xi d\kappa \\
&+ \int_x^b \int_y^d \int_z^f g(\xi,\eta,\kappa) G(x,y,z,\xi,\eta,\kappa) d\eta d\xi d\kappa \\
&+ \int_a^x \int_y^d \int_z^f g(\xi,\eta,\kappa) G(x,y,z,\xi,\eta,\kappa) d\eta d\xi d\kappa \\
&+ \int_a^x \int_c^y \int_z^f g(\xi,\eta,\kappa) G(x,y,z,\xi,\eta,\kappa) d\eta d\xi d\kappa
\end{aligned}$$

$$
\begin{aligned}
&+ \int_x^b \int_c^y \int_z^f g(\xi,\eta,\kappa) G(x,y,z,\xi,\eta,\kappa) d\eta d\xi d\kappa \\
&+ \int_a^x \int_y^d \int_e^z g(\xi,\eta,\kappa) G(x,y,z,\xi,\eta,\kappa) d\eta d\xi d\kappa \\
&= Q_1 + Q_2 + Q_3 + Q_4 + Q_5 + Q_6 + Q_7 + Q_8. \qquad \text{(A.18)}
\end{aligned}
$$

The method of approximation is discussed in [6] for the one-dimensional case. However, the approach for the higher dimensions is exactly the same due to the factorization of the Laplace transform. The basic idea is that the convolution integral I splits to several indefinite convolution integrals representing in total the original integral. The indefinite convolution integrals are approximated by using the spectral properties of the collocation matrices of indefinite integrals in connection with the Laplace transform of the related Green's function. The steps of implementation follow directly the discussion given in [6, 7].

The function **convolutionIntegrals3DS[]** calculates the 3-d convolution integral given in (A.18). To perform the calculation the following input data are needed. First the analytic expression of g is required, second the intervals and their coordinate symbols are needed in the form $\{x_i, a_i, b_i\}$ with $i = 1, 2, 3$. The third set of input data is related to the number of Sinc points in the three different directions M_x, M_y, and M_z. The final parameter is the "width" d to adapt the step length h to an optimal value of the properties of conformal maps. We note that the current implementation is based on the symbolic Laplace transform of the free space Green's function. If there is a need to carry out calculations for other types of Green's functions the specific location in the function must be changed. The next few lines show the implementation. The comments added indicate the steps carried out in the sequel of calculation.

```
(* -- determine the convolution integrals -- *)
(* ---  ∫ax^bx ∫ay^by ∫az^bz G(|r⃗ - r⃗'|) g(x',y',z') dx' dy' dz'  --- *)
(* -- g is a given function -- *)

convolutionIntegrals3DS[ gIn_, { varx_, ax_, bx_}, { vary_, ay_, by_},
{ varz_, az_, bz_}, Mx_, My_, Mz_, d_] :=
Block[{ minpara, α1, k01, Nx, Ny, Nz, hx, hy, hz, sincPointsX, sincPointsY,
sincPointsZ, DmatX, DmatY, DmatZ, AmX, AmY, AmZ, BmX, BmY, BmZ,
evalAX, evalAY, evalAZ, evecAX, evecAY, evecAZ, XAX, XAY, XAZ, XinAX,
XinAY, XinAZ, evalBX, evalBY, evalBZ, evecBX, evecBY, evecBZ, XBX,
XBY, XBZ, XinBX, XinBY, XinBZ, lapA, lapB, hMat, FA, FB, gSincX,
```

```
gSincY, pk, qk, papp, qapp, basis, localKernel, α, k0, ltIn, u, v,
precissionGoal = 120, pmat0, pmat1, ρx, equation1, equation2,
equations, solution, H, lpG, s, t, G, sincPointsXY, Q, H0},


LaunchKernels[];
(* -- fix parameters -- *)
Nx = Mx;
Ny = My;
Nz = Mz;
hx = π/√(d Nx);
hy = π/√(d Ny);
hz = π/√(d Nz);
(* -- generate Sinc points for x, y, and z directions -- *)
sincPointsX = SetPrecision[ Table[ψ[k hx, ax, bx], {k, - Mx, Nx}],
precissionGoal];
sincPointsY =
SetPrecision[ Table[ψ[k hy, ay, by] /. varx ->ψ[k hx, ax, bx], {k, - My, Ny}],
precissionGoal];
sincPointsZ =
SetPrecision[
Table[ψ[k hz, az, bz] /.{ varx ->ψ[k hx, ax, bx], vary ->ψ[k hy, ay, by]},
{k, - Mz, Nz}], precissionGoal];
(* -- generate the discrete representation of the function g
at Sinc points -- *)
gSincX = Map[( gIn /. varx ->#)&, sincPointsX];
(* -- x direction along the row -- *)
gSincX = Map[( gSincX /. vary ->#)&, sincPointsY];
```

```
gSincX = Map[( gSincX /. varz ->#)&, sincPointsZ];
ρx = gSincX;
(* -- Green's function used as kernel -- *)
G = 1/(4π Sqrt[x^2+y^2+z^2]);
(* -- Stengers Formula for the Laplace transform of G -- *)
lpG = lapTr3Poi[s, t, u];
(* -- spectral properties of collocation data -- *)
(* -- diagonal matrix for1/ϕ' numeric -- *)
DmatX =
SetPrecision[
DiagonalMatrix [ Map [(N [1 /∂ varxϕ[ varx, ax, bx] /. varx → #]) &,
sincPointsX]], precissionGoal];
DmatY =
SetPrecision[
DiagonalMatrix[
MapThread [(N [1 /∂ varyϕ[ vary, ay, by] /.{ vary → #2, varx → #1}]) &,
{ sincPointsX, sincPointsY}]], precissionGoal];
DmatZ =
SetPrecision[
DiagonalMatrix[
MapThread[
(N [1 /∂ varzϕ[ varz, az, bz] /.{ varz → #3, vary → #2, varx → #1}]) &,
{ sincPointsX, sincPointsY, sincPointsZ}]], precissionGoal];
(* -- matrixA_m -- *)
AmX = SetPrecision[ hx mT[ Mx, Nx]. DmatX, precissionGoal];
AmY = SetPrecision[ hy mT[ My, Ny]. DmatY, precissionGoal];
AmZ = SetPrecision[ hz mT[ Mz, Nz]. DmatZ, precissionGoal];
```

```
(* -- matrix B_m -- *)
BmX = SetPrecision[ hx Transpose[ mT[ Mx, Nx]]. DmatX, precissionGoal];
BmY = SetPrecision[ hy Transpose[ mT[ My, Ny]]. DmatY, precissionGoal];
BmZ = SetPrecision[ hz Transpose[ mT[ Mz, Nz]]. DmatZ, precissionGoal];
(* -- Eigenvalues and Eigenvectors of Matrix A_m -- *)
{ evalAX, evecAX} = SetPrecision[ Eigensystem[ AmX], precissionGoal];
XAX = Transpose[ evecAX];
XinAX = Inverse[ XAX];
{ evalAY, evecAY} = SetPrecision[ Eigensystem[ AmY], precissionGoal];
XAY = Transpose[ evecAY];
XinAY = Inverse[ XAY];
{ evalAZ, evecAZ} = SetPrecision[ Eigensystem[ AmZ], precissionGoal];
XAZ = Transpose[ evecAZ];
XinAZ = Inverse[ XAZ];
(* -- Eigenvalues and Eigenvectors of Matrix B_m -- *)
{ evalBX, evecBX} = SetPrecision[ Eigensystem[ BmX], precissionGoal];
XBX = Transpose[ evecBX];
XinBX = Inverse[ XBX];
{ evalBY, evecBY} = SetPrecision[ Eigensystem[ BmY], precissionGoal];
XBY = Transpose[ evecBY];
XinBY = Inverse[ XBY];
{ evalBZ, evecBZ} = SetPrecision[ Eigensystem[ BmZ], precissionGoal];
XBZ = Transpose[ evecBZ];
XinBZ = Inverse[ XBZ];
(* end of spectral part *)
(* -- indefinite integrals -- *)
(* -- first integral -- *)
```

```
(* -- integral of the type Q1 = ∫_a^x ∫_a^y ∫_a^z g(ξ,η,κ)G(x,y,z,ξ,η,κ)dηdξdκ -- *)
Print[ "— integral of the type Q1"];
pmat0 = SetPrecision[ matrixConvolInt[ ρx, XinAZ, XinAY, XinAX, lpG,
evalAX, evalAY, evalAZ, XAX, XAY, XAZ], precissionGoal];
(* -- integral of the type Q2 = ∫_x^b ∫_a^y ∫_a^z g(ξ,η,κ)G(x,y,z,ξ,η,κ)dηdξdκ -- *)
Print[ "— integral of the type Q2"];
pmat0 =
pmat0 + SetPrecision[ matrixConvolInt[ ρx, XinAZ, XinAY, XinBX, lpG,
evalBX, evalAY, evalAZ, XBX, XAY, XAZ], precissionGoal];
(* -- integral of the type Q3 = ∫_x^b ∫_y^b ∫_a^z g(ξ,η,κ)G(x,y,z,ξ,η,κ)dηdξdκ -- *)
Print[ "— integral of the type Q3"];
pmat0 =
pmat0 + SetPrecision[ matrixConvolInt[ ρx, XinAZ, XinBY, XinBX, lpG,
evalBX, evalBY, evalAZ, XBX, XBY, XAZ], precissionGoal];
(* -- integral of the type Q4 = ∫_x^b ∫_y^b ∫_z^b g(ξ,η,κ)G(x,y,z,ξ,η,κ)dηdξdκ -- *)
Print[ "— integral of the type Q4"];
pmat0 =
pmat0 + SetPrecision[ matrixConvolInt[ ρx, XinBZ, XinBY, XinBX, lpG,
evalBX, evalBY, evalBZ, XBX, XBY, XBZ], precissionGoal];
(* -- integral of the type Q5 = ∫_a^x ∫_y^b ∫_z^b g(ξ,η,κ)G(x,y,z,ξ,η,κ)dηdξdκ -- *)
Print[ "— integral of the type Q5"];
pmat0 =
pmat0 + SetPrecision[ matrixConvolInt[ ρx, XinBZ, XinBY, XinAX, lpG,
evalAX, evalBY, evalBZ, XAX, XBY, XBZ], precissionGoal];
(* -- integral of the type Q6 = ∫_a^x ∫_a^y ∫_z^b g(ξ,η,κ)G(x,y,z,ξ,η,κ)dηdξdκ -- *)
Print[ "— integral of the type Q6"];
pmat0 =
```

```
pmat0 + SetPrecision[ matrixConvolInt[ ρx, XinBZ, XinAY, XinAX, lpG,
evalAX, evalAY, evalBZ, XAX, XAY, XBZ], precissionGoal];
(* -- integral of the type Q7 = ∫_x^b ∫_a^y ∫_z^b g(ξ,η,κ)G(x,y,z,ξ,η,κ)dηdξdκ -- *)
Print[ "— integral of the type Q7"];
pmat0 =
pmat0 + SetPrecision[ matrixConvolInt[ ρx, XinBZ, XinAY, XinBX, lpG,
evalBX, evalAY, evalBZ, XBX, XAY, XBZ], precissionGoal];
(* -- integral of the type Q8 = ∫_a^x ∫_y^b ∫_a^z g(ξ,η,κ)G(x,y,z,ξ,η,κ)dηdξdκ -- *)
Print[ "— integral of the type Q8"];
pmat0 =
pmat0 + SetPrecision[ matrixConvolInt[ ρx, XinAZ, XinBY, XinAX, lpG,
evalAX, evalBY, evalAZ, XAX, XBY, XAZ], precissionGoal];
(* end of indefinite integral part *)


(* -- create the basis -- *)
basisX = Table[γ[ varx, k, hx, ax, bx], {k, - Mx, Nx}];
basisY = Table[γ[ vary, k, hy, ay, by], {k, - My, Ny}];
basisZ = Table[γ[ varz, k, hz, az, bz], {k, - Mz, Nz}];
(* -- sum up the terms -- *)
apprX = ParallelSum[ pmat0[[j + Mx + 1, k + My + 1, l + Mz + 1]]
basisX[[j + Mx + 1]] basisY[[k + My + 1]] basisZ[[l + Mz + 1]],
{j, - Mx, Mx}, {k, - My, My}, {l, - Mz, Mz}];
(* -- function representation -- *)
jxApr = Piecewise[{{ apprX, ax ≤ varx ≤ bx&& ay ≤ vary ≤ by&& az
                     ≤ varz ≤ bz}}];
(* -- genrate a pure function representation -- *)
solution =
{p → SetPrecision[ Apply[ Function, {{ varx, vary, varz}, $jx}] /. $jx → jxApr,
```

```
$MachinePrecision], sincPointsX, sincPointsY, sincPointsZ, ρx,
{ varx, ax, bx}, { vary, ay, by}, { varz, az, bz}};
(* -- shut down kernels -- *)
CloseKernels[];
(* return results *)
{ apprX, solution}
]
```

In the following application of **convolutionIntegrals3DS[]** we combine some of the functions discussed in the previous examples.

Example A.1.15. 3-d Convolution Integral
The example demonstrates how to solve the initial pressure problem introduced at the beginning of the Appendix. The example is related to the Poisson problem

$$\Delta p = -g \tag{A.19}$$

which is equivalent to a convolution problem for a given function g (see Eq. 1.13). To start with let us first generate the function g derived from a divergence free vector field as initial condition. The initial condition is generated as discussed in Example A.1.1

```
vVectorField = initialValueSelection[zx^2 e^-(x^2+y^2+z^2), zy^2 e^-(x^2+y^2+z^2)]
```

$$\left\{e^{-x^2-y^2-z^2}x^2 - 2e^{-x^2-y^2-z^2}x^2z^2,\right.$$
$$e^{-x^2-y^2-z^2}y^2 - 2e^{-x^2-y^2-z^2}y^2z^2,$$
$$\left.-2e^{-x^2-y^2-z^2}xz + 2e^{-x^2-y^2-z^2}x^3z - 2e^{-x^2-y^2-z^2}yz + 2e^{-x^2-y^2-z^2}y^3z\right\}$$

The graph of the components are shown in Fig. A.3. The plot shows a grid of sections for the initial velocity vector field.

```
vecInitialValuesPlot[ vVectorField, {x, -3, 3}, {y, -3, 3}, {z, -3, 3},
{z → 0, z → 1, y → 0}]
```

The derived vector field can be used to generate the potential function g. We use the function **gh**[] and apply it to the above field to get

```
potentialG = gh[ vVectorField, {x, y, z}] // Simplify
```

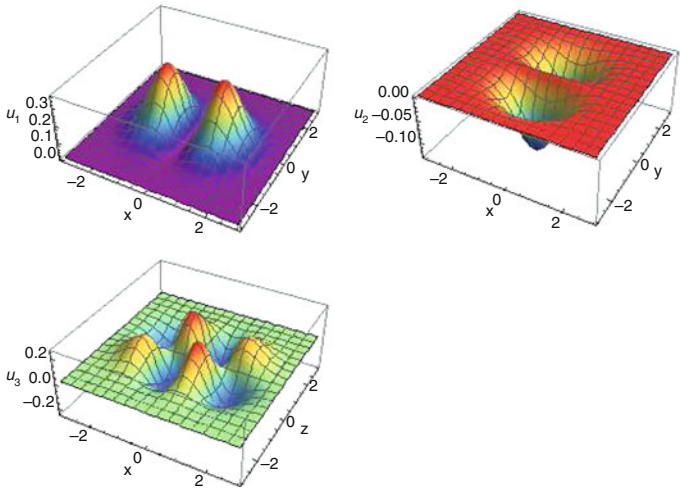

Fig. A.3 Velocity vector fields

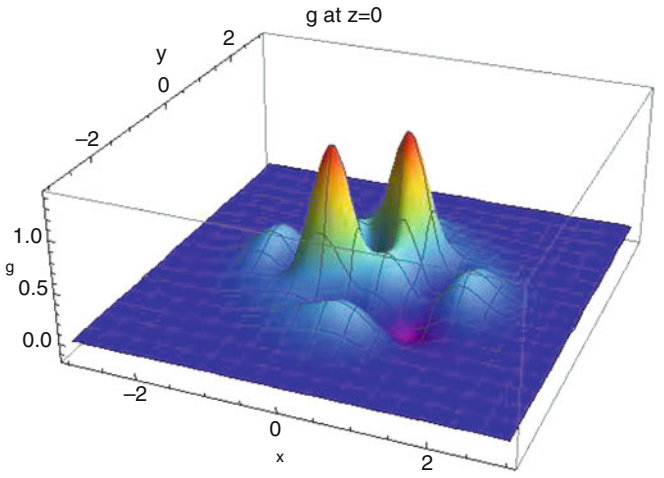

Fig. A.4 Potential g for the pressure

$$8e^{-2\left(x^2+y^2+z^2\right)}\left(x^6\left(1+2z^2\right)+x^3y\left(-1+2y^2\right)\left(1+2z^2\right)+x^4\left(-2-7z^2+2z^4\right)\right.$$
$$+x^2\left(1-z^2+2z^4\right)-xy\left(-\left(1-2z^2\right)^2\right.$$
$$\left.\left.+y^2\left(1+2z^2\right)\right)+y^2\left(1-z^2+2z^4+y^4\left(1+2z^2\right)+y^2\left(-2-7z^2+2z^4\right)\right)\right)$$

The graphical representation of g is shown in Fig. A.4 generated by the following *Mathematics* commands

```
Plot3D[ Evaluate[ potentialG /.z → 0], {x, −3, 3}, {y, −3, 3}, PlotRange → All,
AxesLabel → { x, y, g}, PlotLabel → "g at z=0", ColorFunction → "Rainbow",
PlotPoints → 25]
```

In the next step we apply the Sinc convolution to the derived potential g. The convolution integral using the free space Green's function delivers an approximation of the initial pressure for the N–S equation. We use $N = 24$ Sinc points for the calculation on the total space $\mathbb{R}^3$. For an optimal convergence we choose $d = 2/3$ to adapt the step length accordingly.

```
pressure = convolutionIntegrals3DS[ potentialG, {x, −∞, ∞}, {y, −∞, ∞},
{z, −∞, ∞}, 24, 24, 24, 2/3];
```

— integral of the type Q1
— integral of the type Q2
— integral of the type Q3
— integral of the type Q4
— integral of the type Q5
— integral of the type Q6
— integral of the type Q7
— integral of the type Q8

Figure A.5 shows a 3-d representation of the Sinc convolution result for the pressure. The plot in Fig. A.5 was generated using the following line

```
Plot3D[ Evaluate[ Re[(p[x, y, z] /. First[ Last[ pressure]]) /.z → 0]],
{x, −6, 6}, {y, −6, 6}, ColorFunction → "Rainbow", AxesLabel → { x, y, p},
PlotPoints → 45, PlotRange → All, PlotLabel → "p at z=0, d=2/3, N=24"]
```

The above calculation steps demonstrate that the derivation of a Sinc convolution is straightforward. The calculation needs only the potential function g as an analytic representation. The graph in Fig. A.5 demonstrates that the resulting approximation is smooth.

▽

The above defined function **convolutionIntegrals3DS[]**, although useful and effective, needs some adaption to the N–S algorithm. The N–S algorithm uses 3-d convolutions many times. It is thus beneficial to optimize the calculation for this frequent application of the convolution integral. The next few sections describe how the Sinc convolution is adapted to these issues.

A.1.14 Splitted Convolution Integrals

The effective use of convolution integrals applied to the Navier–Stokes problem needs a modified implementation. The analysis of the convolution algorithm reveals that we can split the computations into two parts. The first part is dealing with the spectral properties and the second one with the discrete Sinc approximation. The first part is always the same for all calculations once the dimension of the approximation N is selected. The second part of the convolution related to the potential g changes. During the calculation the potential changes because the velocity vector field from where g is derived changes. Exactly this is the setup in the application to the N–S problem. To avoid repeated calculation of the first part it is sufficient to calculate the spectral properties only once. In other words the collocation of the domain $\mathscr{D}$ will be always the same. The second part, the Sinc collocation of g, will change in the Gauss–Seidel algorithm. Consequently we split up the convolution calculation into two parts. The first one which will stay always the same is related to the spectral properties. The second one which varies corresponds to the change of g. For this reason we define two functions **spectralProperties3D[]** and **convolutionIntegrals3D[]** which are together equivalent to the function **convolutionIntegrals3DS[]**. The equivalence is established by chaining the spectral part and the variable part. Thus the first spectral part is calculated only once while the second part is used in connection with the results from the first part as often as needed. This approach saves at least the multiple calculation of the spectral properties.

The function **spectralProperties3D[]** is the first part to represent a 3-d convolution integral of the following form

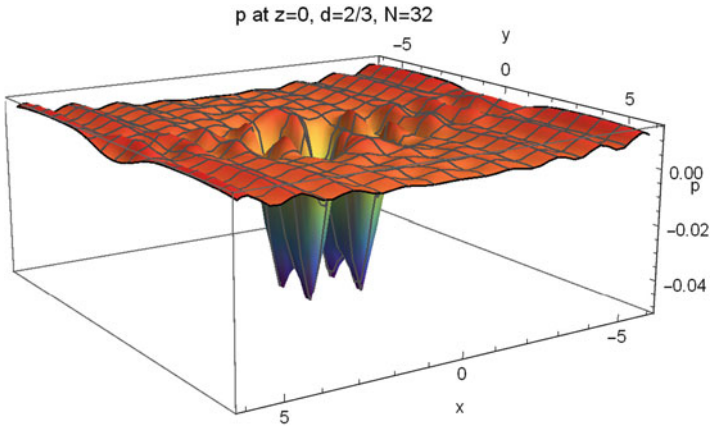

Fig. A.5 Pressure p at $z = 0$. Number of Sinc points $N = 24$

$$\int_a^b \int_c^d \int_e^f G(|\boldsymbol{r} - \boldsymbol{r}'|) g(x', y', z') dx' dy' dz'. \tag{A.20}$$

The function **spectralProperties3D[]** only calculates the spectral data of the discrete representation based on Sinc points in all three directions. The intervals $[a, b]$, $[c, d]$, and $[e, f]$ are in general finite or infinite. For the application to the N–S equation specifically all three intervals are equal to $\mathbb{R}$. Specifically the eigenvalues and eigenvectors are determined for a rectangular geometry. For this reason the boundary points in each of the three coordinate directions are specified as input data. In addition the variables of the three coordinate directions are included in the domain description. The function **spectralProperties3D[]** is based on the free space Green's function $G_0 = 1/(4\pi r)$ which is generated internally of **spectralProperties3D[]**. The Laplace transform of the Green's function is known and calculated based on three functions which we introduce in Sect. A.1.12. Since the spectral properties are frequently used in the solution of N–S equation it is beneficial here to calculate the spectral data only once. The determination of the spectral data and the frequent use of them save time in the iteration process for the N–S equation. The following lines are an implementation of the spectral part.

```
(* -- determine the convolution integrals -- *)
(* --- Integrate[ G[Norm[r - r']] g[x', y', z'], {x', ax, bx}, {y', ay, by}, {z', az, bz}] --- *)
(* -- G is Green's function -- *)
spectralProperties3D[{varx_, ax_, bx_}, {vary_, ay_, by_},
{varz_, az_, bz_}, Mx_, My_, Mz_, hxyz_] :=
Block[{minpara, α1, k01, Nx, Ny, Nz, hx, hy, hz, sincPointsX, sincPointsY,
sincPointsZ, DmatX, DmatY, DmatZ, AmX, AmY, AmZ, BmX, BmY, BmZ,
evalAX, evalAY, evalAZ, evecAX, evecAY, evecAZ, XAX, XAY, XAZ, XinAX,
XinAY, XinAZ, evalBX, evalBY, evalBZ, evecBX, evecBY, evecBZ, XBX,
XBY, XBZ, XinBX, XinBY, XinBZ, lapA, lapB, hMat, FA, FB, gSincX,
gSincY, pk, qk, papp, qapp, basis, localKernel, α, k0, ltIn, u, v,
precissionGoal = 120, pmat0, pmat1, equation1, equation2, equations,
solution, H, lpG, s, t, Q, H0},
(* -- fix parameters -- *)
Nx = Mx;
Ny = My;
```

```
Nz = Mz;
hx = hxyz;
hy = hxyz;
hz = hxyz;
(* -- generate Sinc points -- *)
sincPointsX = SetPrecision[ Table[ψ[k hx, ax, bx], {k, - Mx, Nx}],
precissionGoal];
sincPointsY =
SetPrecision[ Table[ψ[k hy, ay, by] /. varx ->ψ[k hx, ax, bx], {k, - My, Ny}],
precissionGoal];
sincPointsZ =
SetPrecision[
Table[ψ[k hz, az, bz] /.{ varx ->ψ[k hx, ax, bx], vary ->ψ[k hy, ay, by]},
{k, - Mz, Nz}], precissionGoal];
(* -- Stengers formula for the Laplace transform ofG -- *)
lpG = lapTr3Poi[s, t, u];
(* -- diagonal matrix for1/ϕ′ numeric -- *)
DmatX =
SetPrecision[
DiagonalMatrix [ Map [(N [1 /∂ varxϕ[ varx, ax, bx] /. varx → #]) &,
sincPointsX]], precissionGoal];
DmatY =
SetPrecision[
DiagonalMatrix[
MapThread [(N [1 /∂ varyϕ[ vary, ay, by] /.{ vary → #2, varx → #1}]) &,
{ sincPointsX, sincPointsY}]], precissionGoal];
DmatZ =
```

```
SetPrecision[
DiagonalMatrix[
MapThread[
(N[1/∂varz ϕ[varz, az, bz] /.{varz → #3, vary → #2, varx → #1}]) &,
{sincPointsX, sincPointsY, sincPointsZ}]], precissionGoal];
(* -- matrixA_m -- *)
AmX = SetPrecision[hx mT[Mx, Nx].DmatX, precissionGoal];
AmY = SetPrecision[hy mT[My, Ny].DmatY, precissionGoal];
AmZ = SetPrecision[hz mT[Mz, Nz].DmatZ, precissionGoal];
(* -- matrixB_m -- *)
BmX = SetPrecision[hx Transpose[mT[Mx, Nx]].DmatX, precissionGoal];
BmY = SetPrecision[hy Transpose[mT[My, Ny]].DmatY, precissionGoal];
BmZ = SetPrecision[hz Transpose[mT[Mz, Nz]].DmatZ, precissionGoal];
(* -- Eigenvalues and Eigenvectors of MatrixA_m -- *)
{evalAX, evecAX} = SetPrecision[Eigensystem[AmX], precissionGoal];
XAX = Transpose[evecAX];
XinAX = Inverse[XAX];
{evalAY, evecAY} = SetPrecision[Eigensystem[AmY], precissionGoal];
XAY = Transpose[evecAY];
XinAY = Inverse[XAY];
{evalAZ, evecAZ} = SetPrecision[Eigensystem[AmZ], precissionGoal];
XAZ = Transpose[evecAZ];
XinAZ = Inverse[XAZ];
(* -- Eigenvalues and Eigenvectors of MatrixB_m -- *)
{evalBX, evecBX} = SetPrecision[Eigensystem[BmX], precissionGoal];
XBX = Transpose[evecBX];
XinBX = Inverse[XBX];
```

```
{evalBY, evecBY} = SetPrecision[Eigensystem[BmY], precissionGoal];
XBY = Transpose[evecBY];
XinBY = Inverse[XBY];
{evalBZ, evecBZ} = SetPrecision[Eigensystem[BmZ], precissionGoal];
XBZ = Transpose[evecBZ];
XinBZ = Inverse[XBZ];
(* -- returen of the spectral data and the symbolic Green's
function -- *)
{{XAX, XAY, XAZ, XinAX, XinAY, XinAZ, evalAX, evalAY, evalAZ},
{XBX, XBY, XBZ, XinBX, XinBY, XinBZ, evalBX, evalBY, evalBZ}, {lpG}}
]
```

The actual 3-d convolution integral is generated in a second step which is numerously used by the N–S algorithm to represent the spatial component of the convolution integral. The function **convolutionIntegrals3D[]** uses the derived data of the spectrum, the symbolic Laplace representation of the Green's function, and a discrete array including the collocated information of the function g at Sinc points. The discrete representation of g instead of an analytic one is another property of the approach used. It is mandatory in the N–S algorithm to process the discrete representation because during the calculation only discrete arrays are used. The function **convolutionIntegrals3D[]** uses the splitting property of integrals to separate the integration domains in two separated incomplete integrals. This separation results into eight different versions of convolution integrals which are determined one after another to build up the final expression

$$\int_a^b \int_c^d \int_e^f G(|\boldsymbol{r} - \boldsymbol{r}'|) g(x', y', z') dx' dy' dz'. \tag{A.21}$$

The crucial point of this calculation is an effective implementation of array multiplications in 3-d. We use here specifically the parallel properties of *Mathematica* to perform the repeated multiplications and summations in an effective way. It turned out that the implementation of these steps based on matrix multiplications is about ten times slower than the current parallel one. The core steps in this function are the summation of the partial integrals and the calculation of the discrete convolution product based on the Laplace representation. The algorithm for this three-dimensional multiplication is contained in Algorithm 4.6.7 given in [5]. The

implementation of Algorithm 4.6.7 is realized in function **matrixConvolInt[]** which is used eight times with the function **convolutionIntegrals3D[]**.

```
(* -- second part to determine a3 - d convolution integral -- *)
(* --- ∫_{-L/2}^{L/2} ∫_{-L/2}^{L/2} ∫_{-L/2}^{L/2} G(|r⃗ - r⃗'|) g(x',y',z') dx' dy' dz' --- *)
(* -- ρx is a given array and spectrum the spectral data of the
discrete problem -- *)
convolutionIntegrals3D[ spectrum_, ρx_] :=
Block[{ minpara, α1, k01, Nx, Ny, Nz, hx, hy, hz, sincPointsX, sincPointsY,
sincPointsZ, DmatX, DmatY, DmatZ, AmX, AmY, AmZ, BmX, BmY, BmZ,
evalAX, evalAY, evalAZ, evecAX, evecAY, evecAZ, XAX, XAY, XAZ, XinAX,
XinAY, XinAZ, evalBX, evalBY, evalBZ, evecBX, evecBY, evecBZ, XBX,
XBY, XBZ, XinBX, XinBY, XinBZ, lapA, lapB, hMat, FA, FB, gSincX,
gSincY, pk, qk, papp, qapp, basis, localKernel, α, k0, ltIn, u, v,
precissionGoal = 120, pmat0, pmat1, equation1, equation2, equations,
solution, H, lpG, s, t, Q, H0},
(* -- set spectrum to the different matrices and arrays used -- *)
{{ XAX, XAY, XAZ, XinAX, XinAY, XinAZ, evalAX, evalAY, evalAZ},
{XBX, XBY, XBZ, XinBX, XinBY, XinBZ, evalBX, evalBY, evalBZ},{lpG}} =


spectrum;
(* -- distribute definitions to kernels -- *)
DistributeDefinitions[ evalAX, evalAY, evalAZ, evalBX, evalBY, evalBZ, ψ, φ];
(* -- integral -- *)
(* -- I = Q1 + Q2 + Q3 + Q4 + Q5 + Q6 + Q7 + Q8 -- *)
(* -- integral of the type Q1 = ∫_a^x ∫_a^y ∫_a^z g(ξ,η,κ) G(x,y,z,ξ,η,κ) dη dξ dκ -- *)
pmat0 = SetPrecision[ matrixConvolInt[ ρx, XinAZ, XinAY, XinAX, lpG,
evalAX, evalAY, evalAZ, XAX, XAY, XAZ], precissionGoal];
```

```
(* -- integral of the type Q2 = ∫_x^b ∫_a^y ∫_a^z g(ξ,η,κ)G(x,y,z,ξ,η,κ)dηdξdκ -- *)
pmat0 =
pmat0 + SetPrecision[ matrixConvolInt[ ρx, XinAZ, XinAY, XinBX, lpG,
evalBX, evalAY, evalAZ, XBX, XAY, XAZ], precissionGoal];
(* -- integral of the type Q3 = ∫_x^b ∫_y^b ∫_a^z g(ξ,η,κ)G(x,y,z,ξ,η,κ)dηdξdκ -- *)
pmat0 =
pmat0 + SetPrecision[ matrixConvolInt[ ρx, XinAZ, XinBY, XinBX, lpG,
evalBX, evalBY, evalAZ, XBX, XBY, XAZ], precissionGoal];
(* -- integral of the type Q4 = ∫_x^b ∫_y^b ∫_z^b g(ξ,η,κ)G(x,y,z,ξ,η,κ)dηdξdκ -- *)
pmat0 =
pmat0 + SetPrecision[ matrixConvolInt[ ρx, XinBZ, XinBY, XinBX, lpG,
evalBX, evalBY, evalBZ, XBX, XBY, XBZ], precissionGoal];
(* -- integral of the type Q5 = ∫_a^x ∫_y^b ∫_z^b g(ξ,η,κ)G(x,y,z,ξ,η,κ)dηdξdκ -- *)
pmat0 =
pmat0 + SetPrecision[ matrixConvolInt[ ρx, XinBZ, XinBY, XinAX, lpG,
evalAX, evalBY, evalBZ, XAX, XBY, XBZ], precissionGoal];
(* -- integral of the type Q6 = ∫_a^x ∫_a^y ∫_z^b g(ξ,η,κ)G(x,y,z,ξ,η,κ)dηdξdκ -- *)
pmat0 =
pmat0 + SetPrecision[ matrixConvolInt[ ρx, XinBZ, XinAY, XinAX, lpG,
evalAX, evalAY, evalBZ, XAX, XAY, XBZ], precissionGoal];
(* -- integral of the type Q7 = ∫_x^b ∫_a^y ∫_z^b g(ξ,η,κ)G(x,y,z,ξ,η,κ)dηdξdκ -- *)
pmat0 =
pmat0 + SetPrecision[ matrixConvolInt[ ρx, XinBZ, XinAY, XinBX, lpG,
evalBX, evalAY, evalBZ, XBX, XAY, XBZ], precissionGoal];
(* -- integral of the type Q8 = ∫_a^x ∫_y^b ∫_a^z g(ξ,η,κ)G(x,y,z,ξ,η,κ)dηdξdκ -- *)
pmat0 =
pmat0 + SetPrecision[ matrixConvolInt[ ρx, XinAZ, XinBY, XinAX, lpG,
```

```
evalAX, evalBY, evalAZ, XAX, XBY, XAZ], precissionGoal];


pmat0

]
```

The function **matrixConvolInt[]** is the core of the convolution integral functions. It is frequently used in **convolutionIntegrals3D[]** and implements Algorithm 4.6.7 of [5]. This algorithm uses the symbolic representation of the Laplace transform of the Green's function discretized on the eigenvalues of the spatial structure. Matrices of eigenvectors and their inverse matrices in each direction are used to build up the convolution integral in a discrete representation. First the inverse matrices are used in the forward direction of the Algorithm 4.6.7 and then the matrices of the eigenvectors in the backward direction. The midpoint of these matrix multiplications are determined by a Hadarmad product of the discrete Laplace matrix and the forward matrix. We note that the algorithm can be generalized to any dimension as long as the eigenvalues and eigenvectors are known. The separation of the products is also the source of the general separation of coordinate systems [5].

```
(* -- implements Algorithm 4.6.7 by F. Stenger 1993 p. 239 -- *)

matrixConvolInt[ gSinc_, XinAZ_, XinAY_, XinAX_, ltIn_, evalAX_, evalAY_,

evalAZ_, XAX_, XAY_, XAZ_] :=

Block[{ lapA, precissionGoal = 120, m, hh, kk, mm, tt, rr, qq, pp},


(* -- Laplace transform common to all steps -- *)

lapA = SetPrecision[ laplaceTransform3D[ ltIn, evalAX, evalAY, evalAZ],

precissionGoal];

(* -- determine length of the array -- *)

m = Length[ gSinc];

(* -- forward step1 -- *)

hh =

SetPrecision[

ParallelTable[ Sum[ Part[ XinAX, j, μ] Part[ gSinc, μ, k, l], {μ, 1, m}],

{j, 1, m}, {k, 1, m}, {l, 1, m}], precissionGoal];
```

```
(* -- forward step2 -- *)
kk =
SetPrecision[
ParallelTable[ Sum[ Part[ XinAY, k, λ] Part[ hh, j, λ, l], {λ, 1, m}],
{j, 1, m}, {k, 1, m}, {l, 1, m}], precissionGoal];
(* -- forward step3 -- *)
mm =
SetPrecision[
ParallelTable[ Sum[ Part[ XinAZ, l, γ] Part[ kk, j, k, γ], {γ, 1, m}],
{j, 1, m}, {k, 1, m}, {l, 1, m}], precissionGoal];
(* -- Hadamard product step -- *)
tt = SetPrecision[ lapA mm, precissionGoal];
(* -- backward step4 -- *)
rr =
SetPrecision[ ParallelTable[ Sum[ Part[ XAZ, l, ϕ] Part[ tt, j, k, ϕ], {ϕ, 1, m}],
{j, 1, m}, {k, 1, m}, {l, 1, m}], precissionGoal];
(* -- backward step5 -- *)
qq =
SetPrecision[ ParallelTable[ Sum[ Part[ XAY, k, κ] Part[ rr, j, κ, l], {κ, 1, m}],
{j, 1, m}, {k, 1, m}, {l, 1, m}], precissionGoal];
(* -- backward step6 -- *)
pp =
SetPrecision[ ParallelTable[ Sum[ Part[ XAX, j, α] Part[ qq, α, k, l], {α, 1, m}],
{j, 1, m}, {k, 1, m}, {l, 1, m}], precissionGoal];

pp
]
```

The two functions of a discrete Sinc convolution are effectively used in the main function **NavierStokes3D[]**.

A.1.15 Navier–Stokes Function

The function **NavierStokes3D[]** is the main function to approximate the Navier–Stokes initial-boundary value problem based on divergence free initial vector fields. The function is setup to use in a way the analytic initial conditions as input data. Thus function **NavierStokes3D[]** uses a list of analytic initial values for the different directions of the initial velocities $u^{0,j}$ with $j = 1, 2, 3$. Each spatial direction of the domain is specified with its lower and upper limits a_j and b_j. In addition to the terminal values the related independent variables are specified for the spatial direction, respectively. The same list structure is used for the temporal variable $t \in [a_4, b_4]$. The number of Sinc points N for the spatial and M for the temporal direction is specified next. The spatial step length parameter δ allows to adapt the spatial Sinc step length in an optimal way. The parameter ϵ, the viscosity of the fluid, related to the scaling of the time axis is given next. The expected error level for the iteration of the Gauss–Seidel iteration is the last mandatory input data value. The function **NavierStokes3D[]** carries out the explicit calculations for the nonlinear integral equations specified in the algorithm given in Chap. 5. Whenever possible the steps of the algorithm are performed in parallel to increase the performance of the computation. The efficiency of the function depends on the available kernels on a machine or on a network. The tests were ruining on single machines with two up to eight kernels and on a network with 22 kernels. In addition we made some test runs on a highly parallel grid of an HPC computer. The general observation for the notebook network was that the calculations on a single machine performed better than on a network of machines. The reason is the extensive exchange of data among the kernels which is better organized on board than over a network. The HPC computations performed in this respect in a better way.

The function **NavierStokes3D[]** uses the option **exportResultsTo** to specify a storage location for results and input data. At the end of the calculation the complete set of generated results, symbolic and numeric, and the input data are written to the specified file. The preservation of results avoids to run time consuming calculations again and again. In addition the storage of data allows a post-processing of the approximation in other applications.

The next lines list the implementation of the main function **NavierStokes3D[]**. The comments given in the program code describe the current calculation step performed.

```
Options[ NavierStokes3D] = { exportResultsTo → { "Results_001.m"}};
```

```
NavierStokes3D[ initial_, { varx_, ax_, bx_}, { vary_, ay_, by_},
{ varz_, az_, bz_}, { vart_, at_, bt_}, Mx_, Mt_, δ_, ϵ_, errorLevel_,
OptionsPattern[]] :=
Block[{ precissionGoal = 20, My, Mz, n, h, hp, tt, hμ, sincPoints, Dmat,
Am, spectrum3D, u1, u2, u3, Fu1, Fu2, Fu3, λb, FV1, FV2, FV3, V1, V2,
V3, W1, W2, W3, FW1, FW2, FW3, FU1, FU2, FU3, X1, X2, X3, X4, X5, X6,
FX1, FX2, FX3, FX4, FX5, FX6, J, K, L, Ff, f, Fp, p, Y, Z, FN, U1, U2,
U3, Δ1, Δ2, Δ3, errors, er0 = 1, UU1, UU2, UU3, pP, hs, hcon, eigenValuesT,
eigenVectorsT, XAT, XinAT, Tstep, CT, d, exportData, fileName},
(* -- set the file name for exporting results fileName -- *)
fileName = OptionValue[ exportResultsTo];
(* *)
(* -- start parallel kernels -- *)
LaunchKernels[];
(* -- define step lengths -- *)
My = Mx;
Mz = Mx;
n = Mx;
(* -- spatial step length for convolutions -- *)
hcon = π/√δn;
(* -- determined for Sinc collocation in Fourier data -- *)
d = d /. Flatten[ Solve[ (bx - ax)/(d Mx) == hcon, d]];
h = (bx - ax)/(d Mx);
(* -- hp = h' used in Fourier steps -- *)
hp = π/(nh);
(* -- create discrete array for Λ -- *)
λb = Λ[n, hp];
```

```
(* -- choose the step length for the time line -- *)
hμ = π/√(13 Mt);
(* -- Sinc points along the time axis -- *)
tt = N[ ParallelTable[ bt Exp[μ hμ]/(1+ Exp[μ hμ]), {μ, − Mt, Mt}]];
(*
Print[ "h = ", N[h]];
Print[ "hcon = ", N[ hcon]];
Print[ "d = ", N[d]];
Print[ "hp = ", N[ hp]];
Print[ "hμ = ", N[ hμ ]];
*)
(* -- create the sinc points on the time line -- *)
sincPoints = SetPrecision[ Table[ψ[k hμ, at, bt], {k, − Mt, Mt}],
precissionGoal];
(* -- define the diagonal matrix D for the time axis -- *)
Dmat = DiagonalMatrix[
Map [(N [1 /∂ vart φ[ vart, at, bt] /. vart → #, precissionGoal]) &,
sincPoints]];
(* -- define the integral operator as a matrix -- *)
Am = hμ mT[ Mt, Mt]. Dmat;
(* -- calculate spectral properties of the time coordinate -- *)
{ eigenValuesT, eigenVectorsT} = Eigensystem[ Am];
XAT = Transpose[ eigenVectorsT];
XinAT = Inverse[ XAT];
(* *)
(* -- temporal evolution included in two matrices Tstep and CT -- *)
(* -- matrix d_μ of step b) is generated -- *)
```

```
Tstep = ParallelTable[ Part[ eigenValuesT, μ] / (1 + ϵ Part[ λb, j, k, l]^2 Part[ eigenValuesT, μ]),
{j, 1, 2n + 1}, {k, 1, 2n + 1}, {l, 1, 2n + 1}, {μ, 1, 2 Mt + 1}];
(* Print[ "Dim Tstep = ", Dimensions[ Tstep]]; *)
(* -- matrixC of stepc) is generated -- *)
CT = ParallelTable[
Sum[ Part[ XAT, κ, μ] Part[ Tstep, j, k, l, μ] Part[ XinAT, μ, λ],
{μ, 1, 2 Mt + 1}], {j, 1, 2n + 1}, {k, 1, 2n + 1}, {l, 1, 2n + 1},
{κ, 1, 2 Mt + 1}, {λ, 1, 2 Mt + 1}];
(* Print[ "- CT done --- "];
Print[ "Dim CT = ", Dimensions[ CT]]; *)

(* -- calculate the spectral properties of the spatial
coordinates. It is important to note that we use equidistant
spacing -- *)
spectrum3D = spectralProperties3D[{ varx, -∞, ∞}, { vary, -∞, ∞},
{ varz, -∞, ∞}, Mx, My, Mz, hcon];
Print[ "- Spectrum done --- "];
(* -- generate the discrete representation of initial values -- *)
u1 = discreteInitialValue[ initial, 1, n, h];
u2 = discreteInitialValue[ initial, 2, n, h];
u3 = discreteInitialValue[ initial, 3, n, h];
(* -- Fourier transform the initial values by DFT -- *)
Fu1 = AbsoluteTiming[ a3DFourier[ u1, h, n]];
(* Print[ "Fourier time 3D = ", First[ Fu1]]; *)
Fu1 = Last[ Fu1];
Fu2 = a3DFourier[ u2, h, n];
Fu3 = a3DFourier[ u3, h, n];
```

```
(* -- determine the4 - d convolution of initial data and the
time line in Fourier space -- *)
FV1 = convol4D[ λb, tt, Fu1, ϵ, n];
FV2 = convol4D[ λb, tt, Fu2, ϵ, n];
FV3 = convol4D[ λb, tt, Fu3, ϵ, n];
(* Print[ "Dim FV1 = ", Dimensions[ FV1]]; *)
(* -- Calculate the inverse discrete Fourier transform( DFT)
of the convolution -- *)
V1 = Map[ Chop[ inverseB3DFourier[#, h, n]]&, FV1];
(* Print[ "Dim V1 mapped = ", Map[ Dimensions[#]&, FV1]]; *)
V2 = Map[ Chop[ inverseB3DFourier[#, h, n]]&, FV2];
V3 = Map[ Chop[ inverseB3DFourier[#, h, n]]&, FV3];
(* Print[ "Dim V1 = ", Dimensions[ V1]]; *)
(* -- set variables as initials for the iteration -- *)
W1 = V1;
W2 = V2;
W3 = V3;

FW1 = FV1;
FW2 = FV2;
FW3 = FV3;

U1 = W1;
U2 = W2;
U3 = W3;
(* -- here starts the While LOOP -- *)
While[ er0 > errorLevel,
```

```
Print[ "– enter Loop — "];
FU1 = FW1;
FU2 = FW2;
FU3 = FW3;

(* – – form the Hadarmad products – – *)
X1 = ( U1 U1);
X2 = ( U1 U2);
X3 = ( U1 U3);
X4 = ( U2 U2);
X5 = ( U2 U3);
X6 = ( U3 U3);
(* Print[ "Dim X1 = ", Dimensions[ X1]]; *)
(* – – Fourier transform of each time step and each product – – *)
FX1 = Map[ a3DFourier[#, h, n]&, X1];
FX2 = Map[ a3DFourier[#, h, n]&, X2];
FX3 = Map[ a3DFourier[#, h, n]&, X3];
FX4 = Map[ a3DFourier[#, h, n]&, X4];
FX5 = Map[ a3DFourier[#, h, n]&, X5];
FX6 = Map[ a3DFourier[#, h, n]&, X6];
Print[ "– Fourier I done — "];

(* *)

(* – – create arrays representing λ^j factors – – *)
J = ParallelTable[j hp, {j, −n, n}, {k, −n, n}, {l, −n, n}];
K = ParallelTable[k hp, {j, −n, n}, {k, −n, n}, {l, −n, n}];
```

```
L = ParallelTable[l hp, {j, -n, n}, {k, -n, n}, {l, -n, n}];


(* Print[ "Dim FX1 =", Dimensions[ FX1]]; *)
(* -- form the gradient of the velocities in Fourier space -- *)
Ff =
-( Chop[( Map[JJ#&, FX1] + Map[2JK#&, FX2] + Map[2JL#&, FX3])+
( Map[KK#&, FX4] + Map[2KL#&, FX5] + Map[LL#&, FX6])]);
(* -- inverse DFT of the velocity gradients -- *)
f = Map[ Chop[ inverseB3DFourier[#, h, n]]&, Ff];
(* -- convolution of the spectral data in space and the velocity
gradients delivers the pressure -- *)
Print[ "- convolution I — "];
p = AbsoluteTiming[ Map[ convolutionIntegrals3D[ spectrum3D, #]&, f]];
Print[ "— parallel time convol I = ", First[p]];
p = Last[p];
Print[ "- convolution I done — "];
(* -- DFT of the pressure -- *)
Fp = Map[ a3DFourier[#, h, n]&, p];
(* -- form the parts of the integral terms in Fourier space -- *)
Y = Map[(iJ#)&, Fp];
Z = i Chop[( Map[J#&, FX1] + Map[K#&, FX2] + Map[L#&, FX3])];
FN = Y + Z;
(* *)
(* -- time step of the integrals -- *)
FN = ParallelTable[ Sum[ Part[ CT, j, k, l, μ, m] Part[ FN, μ, j, k, l],
{μ, 1, 2 Mt + 1}], {m, 1, 2 Mt + 1}, {j, 1, 2n + 1}, {k, 1, 2n + 1},
{l, 1, 2n + 1}];
```

```
Print[ "– time step I done — "];
(* – – form the DFT of the first component of the velocity field – – *)
FU1 = FV1 + FN;
U1 = Chop[ Map[ inverseB3DFourier[#, h, n]&, FU1]];
(* – – update the matrices for the second equation – – *)
X1 = ( U1 U1);
X2 = ( U1 U2);
X3 = ( U1 U3);
(* – – Fourier transform the arrays – – *)
FX1 = Map[ a3DFourier[#, h, n]&, X1];
FX2 = Map[ a3DFourier[#, h, n]&, X2];
FX3 = Map[ a3DFourier[#, h, n]&, X3];
Print[ "– Fourier II done — "];
(* – – form the gradient of the velocities in Fourier space for
equation two – – *)
Ff =
−( Chop[( Map[JJ#&, FX1] + Map[2JK#&, FX2] + Map[2JL#&, FX3])+
( Map[KK#&, FX4] + Map[2KL#&, FX5] + Map[LL#&, FX6])]);
(* – – inverse DFT of the velocity gradients for equation two – – *)
f = Map[ Chop[ inverseB3DFourier[#, h, n]]&, Ff];
(* – – convolution of the spectral data in space and the velocity
gradients delivers the pressure for equation two – – *)
Print[ "– convolution II — "];
p = AbsoluteTiming[ Map[ convolutionIntegrals3D[ spectrum3D, #]&, f]];
Print[ "– parallel time =", First[p]];
p = Last[p];
Print[ "– convolution II done — "];
```

```
(* -- DFT of the pressure -- *)
Fp = Map[ a3DFourier[#, h, n]&, p];
(* -- form the parts of the integral terms in Fourier space -- *)
Y = Map[(iK#)&, Fp];
Z = i Chop[( Map[J#&, FX2] + Map[K#&, FX4] + Map[L#&, FX5])];
FN = Y + Z;
(* -- time step of the integrals -- *)
FN = ParallelTable[ Sum[ Part[ CT, j, k, l, μ, m] Part[ FN, μ, j, k, l],
{μ, 1, 2 Mt + 1}], {m, 1, 2 Mt + 1}, {j, 1, 2n + 1}, {k, 1, 2n + 1},
{l, 1, 2n + 1}];
Print[ "- time step II done --- "];
FU2 = FV2 + FN;
U2 = Chop[ Map[ inverseB3DFourier[#, h, n]&, FU2]];
(* -- update the matrices for the third equation -- *)
X2 = ( U1 U2);
X4 = ( U2 U2);
X5 = ( U2 U3);
(* -- Fourier transform the arrays -- *)
FX2 = Map[ a3DFourier[#, h, n]&, X2];
FX4 = Map[ a3DFourier[#, h, n]&, X4];
FX5 = Map[ a3DFourier[#, h, n]&, X5];
Print[ "- Fourier III done --- "];
(* -- form the gradient of the velocities in Fourier space for
equation three -- *)
Ff =
-( Chop[( Map[JJ#&, FX1] + Map[2JK#&, FX2] + Map[2JL#&, FX3])+
( Map[KK#&, FX4] + Map[2KL#&, FX5] + Map[LL#&, FX6])]);
```

```
(* -- inverse DFT of the velocity gradients for equation three -- *)
f = Map[ Chop[ inverseB3DFourier[#, h, n]]&, Ff];
Print[ "- convolution III —— "];
p = AbsoluteTiming[ Map[ convolutionIntegrals3D[ spectrum3D, #]&, f]];
Print[ "- parallel time =", First[p]];
p = Last[p];
Print[ "- convolution III done —— "];
(* -- DFT of the pressure -- *)
Fp = Map[ a3DFourier[#, h, n]&, p];
(* -- form the parts of the integral terms in Fourier space -- *)
Y = Map[(iL#)&, Fp];
Z = i Chop[( Map[J#&, FX3] + Map[K#&, FX5] + Map[L#&, FX6])];
FN = Y + Z;
(* -- time step of the integrals -- *)
FN = ParallelTable[ Sum[ Part[ CT, j, k, l, μ, m] Part[ FN, μ, j, k, l],
{μ, 1, 2 Mt + 1}], {m, 1, 2 Mt + 1}, {j, 1, 2n + 1}, {k, 1, 2n + 1},
{l, 1, 2n + 1}];
Print[ "- time step III done —— "];
FU3 = FV3 + FN;
(* -- determine the last component of the velocity field -- *)
U3 = Chop[ Map[ inverseB3DFourier[#, h, n]&, FU3]];
(* -- determine the differences between the initial step and
the current step -- *)
Δ1 = W1 - U1;
(* Print[ "Dim Δ1 = ", Dimensions[ Δ1]]; *)
Δ2 = W2 - U2;
Δ3 = W3 - U3;
```

```
(* *)


(* -- calculate the errors based on Frobenius matrix norms -- *)
errors =
{ Norm[ Map[ Norm[ Map[ Norm[#, "Frobenius"]&, #], "Frobenius"]&, Delta1],
"Frobenius"]^2,
Norm[ Map[ Norm[ Map[ Norm[#, "Frobenius"]&, #], "Frobenius"]&, Δ2],
"Frobenius"]^2,
Norm[ Map[ Norm[ Map[ Norm[#, "Frobenius"]&, #], "Frobenius"]&, Δ3],
"Frobenius"]^2} ;
Print [" errors^2 = ", errors] ;
er0 = Fold[ Plus, 0, errors]^(1/2);
Print[ "error sum = ", er0];
(* -- set new initials and their DFTs -- *)
W1 = U1;
W2 = U2;
W3 = U3;
FW1 = Map[ a3DFourier[#, h, n]&, W1];
FW2 = Map[ a3DFourier[#, h, n]&, W2];
FW3 = Map[ a3DFourier[#, h, n]&, W3]
];
(* -- create the solutions as analytic representations -- *)
(* -- velocity u_1 -- *)
UU1 =
uh1 → Apply[ Function, {{ varx, vary, varz, vart},
ParallelSum[ Part[ U1, μ + Mt + 1, j + n + 1, k + n + 1, l + n + 1]
Sinc[π( varx - jh)/h] Sinc[π( vary - kh)/h] Sinc[π( varz - lh)/h]
```

```
Sinc[π(φ[ vart, at, bt] − μ hμ)/ hμ], {j, −n, n}, {k, −n, n},
{l, −n, n}, {μ, − Mt, Mt}]];
(* −− velocity u2 −− *)
UU2 =
uh2 → Apply[ Function, {{ varx, vary, varz, vart},
ParallelSum[ Part[ U2, μ + Mt + 1, j + n + 1, k + n + 1, l + n + 1]
Sinc[π( varx − jh)/h] Sinc[π( vary − kh)/h] Sinc[π( varz − lh)/h]
Sinc[π(φ[ vart, at, bt] − μ hμ)/ hμ], {j, −n, n}, {k, −n, n},
{l, −n, n}, {μ, − Mt, Mt}]];
(* −− velocity u3 −− *)
UU3 =
uh3 → Apply[ Function, {{ varx, vary, varz, vart},
ParallelSum[ Part[ U3, μ + Mt + 1, j + n + 1, k + n + 1, l + n + 1]
Sinc[π( varx − jh)/h] Sinc[π( vary − kh)/h] Sinc[π( varz − lh)/h]
Sinc[π(φ[ vart, at, bt] − μ hμ)/ hμ], {j, −n, n}, {k, −n, n},
{l, −n, n}, {μ, − Mt, Mt}]];
(* −− pressure −− *)
pP =
ph → Apply[ Function, {{ varx, vary, varz, vart},
ParallelSum[ Part[p, μ + Mt + 1, j + n + 1, k + n + 1, l + n + 1]
Sinc[π( varx − jh)/h] Sinc[π( vary − kh)/h] Sinc[π( varz − lh)/h]
Sinc[π(φ[ vart, at, bt] − μ hμ)/ hμ], {j, −n, n}, {k, −n, n},
{l, −n, n}, {μ, − Mt, Mt}]];
(* −− shut down kernels −− *)
CloseKernels[];
(* −− export results to a file −− *)
exportData =
```

```
{{ initial, { varx, ax, bx}, { vary, ay, by}, { varz, az, bz},
{ vart, at, bt}, Mx, Mt, δ, ϵ, errorLevel},
{ errors, UU1, UU2, UU3, pP, { U1, U2, U3, p}}};
If[ Length[ fileName] ≥ 1, Print[ "+++ writing data —"];
Export[ fileName, exportData];
Print[ "— writing done +++"]; ];
(* – – return results – – *)
{ errors, UU1, UU2, UU3, pP, { U1, U2, U3, p}}
]
```

The following example demonstrates how **NavierStokes3D[]** is used to solve an N–S initial boundary value problem.

Example A.1.16. Solution of the N–S Problem
The application of **NavierStokes3D[]** to the initial velocity field **vVectorField** (generated in Example A.1.15), on a cube $\mathscr{D} = (-4, 4)^3$ with $N = 8$ and $M = 3$, $\delta = 2/3$, $\varepsilon = 0.5$, and an error margin of 10^{-2} delivers an approximation

```
nSsolution = NavierStokes3D[ vVectorField, {x, −4, 4}, {y, −4, 4},
{z, −4, 4}, {t, 0, 10^−8} , 8, 3, 2/3, .5, 0.01,
exportResultsTo → { "Results_001.m"}]
```

– Spectrum done —
– enter Loop —
– Fourier I done —
– convolution I —
— parallel time convol I = 1666.458569
– convolution I done —
– time step I done —
– Fourier II done —
– convolution II —
– parallel time =2041.654744
– convolution II done —
– time step II done —
– Fourier III done —
– convolution III —
– parallel time =1848.242868
– convolution III done —

– time step III done —
errors^2 = { 9.37351258276783`*^-9,
9.627222913001091`*^-9, 8.4431997499249`*^-9}
error sum = 0.000165662

large output | show less | show more | show all | set size limit...

The result of the calculation is a large list of data consisting mainly of six elements assigned to the symbol **nSsolution**. The structure of the list has the following form

$$\{\varepsilon, u^1, u^2, u^3, p, \{\,\mathrm{U1},\ \mathrm{U2},\ \mathrm{U3}, P\}\}\,. \tag{A.22}$$

where the entries have the meaning:

1. ε represents a list of square errors based on the Frobenius norm for matrices
2. u^1, u^2, u^3, and p are analytic representations of the approximations for the components of the velocity vector field and the pressure. The functions are represented as pure *Mathematica* functions using replacement rules.
3. U1, U2, U3, and P are the discrete numeric results (expansion coefficients) of the approximation.

To access the elements of the results list **nSsolution** we use the **Part[]** function of *Mathematica*. For example the first part of the results list contains the errors squared which are accessed by

```
nSsolution[[1]]
```

```
{ 9.37351258276783`*^-9, 9.627222913001091`*^-9, 8.4431997499249`*^-9}
```

Note the errors are extremely small for using only $m = 17$ Sinc points.

The approximations of the vector field components u^i and the pressure are used in applications as replacement rules. The components u^i use the specific representation **uh1[x, y, z, t]** and similar patterns for u^2, u^3, and p. The symbol **uh1** is replaced (**/.**) by the analytic expression contained in the second part of **nSsolution[[2]]**. The following plots collect in a graphics grid the four approximations at a fixed time $t/T = 0.25$ (see Fig. A.6).

```
GraphicsGrid[
{{ Plot3D [ Evaluate [ Re [ uh1 [x, y, 0, 0.25 10^-8]]] /. nSsolution[[2]]],
{x, -4, 4}, {y, -4, 4}, PlotRange → All, AxesLabel → { x, y, " "},
PlotLabel → "Re(u1 ) at z=0, t/T=0.25", ColorFunction → "Rainbow",
PlotPoints → 50],
```

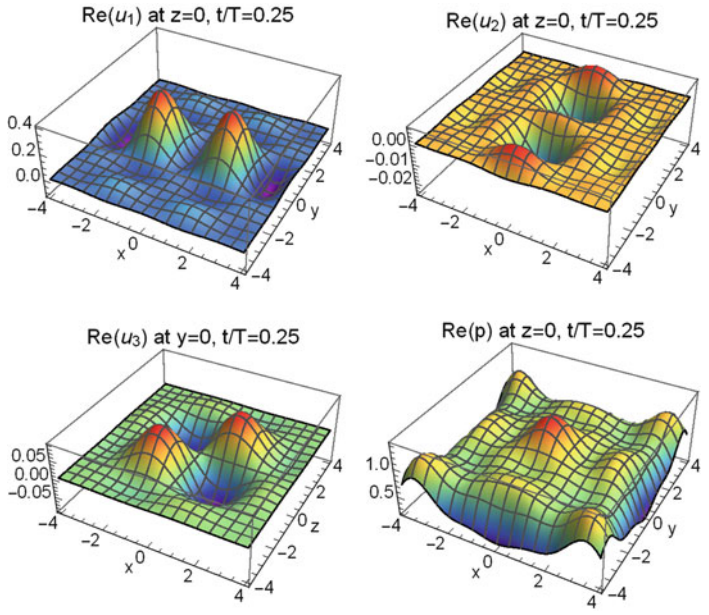

Fig. A.6 Approximations for the vector fields and the pressure

```
Plot3D[ Evaluate[ Re[ uh2[x, y, 1, 0.25 10^-8]] /. nSsolution[[3]]],
{x, -4, 4}, {y, -4, 4}, PlotRange → All, AxesLabel → { x, y, " "},
PlotLabel → "Re(u2 ) at z=0, t/T=0.25", ColorFunction → "Rainbow",
PlotPoints → 50]},
{ Plot3D[ Evaluate[ Re[ uh3[x, 0, y, 0.25 10^-8]] /. nSsolution[[4]]],
{x, -4, 4}, {y, -4, 4}, PlotRange → All, AxesLabel → { x, z, " "},
PlotLabel → "Re(u3 ) at y=0, t/T=0.25", ColorFunction → "Rainbow",
PlotPoints → 50],
Plot3D[ Evaluate[ Re[ ph[x, y, 0, 0.25 10^-8]] /. nSsolution[[5]]],
{x, -4, 4}, {y, -4, 4}, PlotRange → All, AxesLabel → { x, y, " "},
PlotLabel → "Re(p) at z=0, t/T=0.25", ColorFunction → "Rainbow",
PlotPoints → 50]}]
```

The graphs in Fig. A.6 show the real part of the analytic approximations. Due to the small number of Sinc points used in the calculation, we observe ripples of

the velocities at the boundaries. The pressure defined by Eq. (1.13) is a derived quantity. In addition the pressure is calculated based on derivatives which increases the sensitivity. The conclusion at this stage of calculation is that an increase of Sinc point is needed to resolve a stable approximation. We note that the approximations are in general complex valued functions. However, the imaginary part of the approximation is very small and thus insignificant.
▽

A.1.16 Import Function

The function **NavierStokes3D[]** uses an option to store the results of the calculation as symbolic expressions for all the vector field components. Once stored the data are retrieved for additional calculations by the function **importResults[]**. The read in data structure is exactly the same as generated by **NavierStokes3D[]**. The data are organized in two sub-lists. The first collects all the initial data to start a calculation and thus offers the option to rerun a calculation. The second part of the list collects the results of the approximation. All the vector field components of the velocity field and the pressure are available as symbolic expressions. In addition the discrete representations (expansion coefficients) for the vector field components and the pressure are also included in this list of results.

```
importResults[ fileName_] :=
Block[{ initial, varx, ax, bx, vary, ay, by, varz, az, bz, vart, at,
bt, Mx, Mt, δ, ϵ, errorLevel, errors, UU1, UU2, UU3, pP, U1, U2, U3, p},
(* -- import data structure -- *)
{{ initial, { varx, ax, bx}, { vary, ay, by}, { varz, az, bz},
{ vart, at, bt}, Mx, Mt, δ, ϵ, errorLevel},
{ errors, UU1, UU2, UU3, pP, { U1, U2, U3, p}}} = Import[ fileName]
]
```

The use of stored results generated by **NavierStokes3D[]** is straightforward. The function **importResults[]** will read in the data from the specified file. The next example demonstrates the application of the function.

Example A.1.17. Import of N–S Calculations
The current example reads in the data generated in Example A.1.16 which was saved in file **Results_001.m**. The structure of the input is assigned to a symbol, here **solutionNS**, containing the input and output data of **NavierStokes3D[]**.

solutionNS = importResults["Results_001.m"]

large output | show less | show more | show all | set size limit...

The first part of the list contains the input data

inputData = First[solutionNS]

$$\left\{\left\{e^{-x^2-y^2-z^2}x^2 - 2e^{-x^2-y^2-z^2}x^2z^2,\ e^{-x^2-y^2-z^2}y^2 - 2e^{-x^2-y^2-z^2}y^2z^2,\right.\right.$$
$$\left.\left.-2e^{-x^2-y^2-z^2}xz + 2e^{-x^2-y^2-z^2}x^3z - 2e^{-x^2-y^2-z^2}yz + 2e^{-x^2-y^2-z^2}y^3z\right\},\right.$$
$$\{x,-4,4\},\{y,-4,4\},\{z,-4,4\},\left\{t,0,\tfrac{1}{100000000}\right\},8,3,\tfrac{2}{3},0.5,0.01\}$$

The second part collects the results of the calculation. Here shown as shortened list.

nSsolution = Last[solutionNS]

large output | show less | show more | show all | set size limit...

▽

Appendix B
Result Notes

Abstract The following section includes an example of results generated by **NavierStokes3D[]**.

B.1 Example File of Results

The following listing of the output of **NavierStokes3D[]** contains two parts. The first part represents the input data for **NavierStokes3D[]** and the second part contains the results generated by **NavierStokes3D[]** (see the listing below). The listing starts with a comment line generated automatically by *Mathematica*. The next three lines are the input data. The rest of the data are results generated by **NavierStokes3D[]**. The vector components (u^1, u^2, u^3) and p are represented as pure functions using symbols $(uh1, uh2, uh3)$ and ph, respectively. The numeric values at the end of the listing represent the discrete coefficients for the Sinc approximations of the vector field components and the pressure.

The result of the calculation is a list consisting of six elements. The structure of the list has the following global form

$$\left\{\varepsilon, u^1, u^2, u^3, p, \{\,\mathrm{U1},\ \mathrm{U2},\ \mathrm{U3}, P\}\right\}. \tag{B.1}$$

where the entries have the meaning:

1. ε represents a list of square errors based on the Frobenius norm of matrices
2. u^1, u^2, u^3, and p are analytic representations of the approximations for the components of the velocity vector field and the pressure. The functions are represented as pure *Mathematica* functions using replacement rules.
3. U1, U2, U3, and P are the discrete results (expansion coefficients) of the approximation.

```
(* Created with the Wolfram Language : www.wolfram.com *)
{{{E^(-x^2 - y^2 - z^2)*(y - z), E^(-x^2 - y^2 - z^2)*(-x + z),
   E^(-x^2 - y^2 - z^2)*(x - y)}, {x, -3, 3}, {y, -3, 3}, {z, -3, 3},
  {t, 0, 1.*^-6}, 2, 1, 0.5, 1/100},
 {{2.00427284873504 46*^-9, 2.00427283593837*^-9, 2.004272823975107*^-9},
  uh1 -> Function[{x, y, z, t}, -1.720188105222856*^-6*Sinc[2*(-Pi/2 + x)]*
      Sinc[2*y]*Sinc[2*z]*Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     1.720188105222856*^-6*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*z]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 4.244042763168937*^-6*
      Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 4.244042763168939*^-6*
      Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.19426493698813607*Sinc[2*x]*
```

© Springer International Publishing AG 2016

F. Stenger et al., *Navier–Stokes Equations on* $\mathbb{R}^3 \times [0, T]$,
DOI 10.1007/978-3-319-27526-0

```
 Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
0.03340600713028449*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
(0.019752908736525097 - 1.3803662110999904*^-7*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
(0.01975772668211167 + 1.3803662111522862*^-7*I)*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
0.03340600713028449*Sinc[2*(Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.19426493698813607*Sinc[2*x]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*z]*Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
0.03340600713028449*Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
(0.01975772668211167 + 1.3803662110999904*^-7*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*z]*Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
(0.019752908736525097 - 1.3803662111522862*^-7*I)*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*z]*Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
0.03340600713028449*Sinc[2*(Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 4.2440427631689337*^-6*
 Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 4.2440427631689339*^-6*
 Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 4.2440427631618842*^-6*
 Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 4.2440427631618415*^-6*
 Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 6.9091114514986625*^-6*
 Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 6.9091114514986625*^-6*
 Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.033406003792346574*Sinc[2*x]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.005744537441177679*
 Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
(0.00339408495296515577 - 1.380366211067981*^-7*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
(0.003400207226272135 + 1.380366211062865*^-7*I)*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.005744537441177679*
 Sinc[2*(Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.033406003792346574*Sinc[2*x]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.005744537441177679*
 Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
(0.003400207226272135 + 1.380366211067981*^-7*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
(0.00339408495296515577 - 1.3803662110628654*^-7*I)*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.005744537441177679*
 Sinc[2*(Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 6.9091114514986625*^-6*
 Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 6.9091114514986625*^-6*
 Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.19426493431552047*Sinc[2*x]*
 Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
0.03340600104954632*Sinc[2*(-Pi + x)]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
(0.019756074331108424 - 1.3803662110765446*^-7*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
(0.019754558322268188 + 1.3803662110369542*^-7*I)*Sinc[2*(Pi/2 + x)]*
 Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
0.03340600104954632*Sinc[2*(Pi + x)]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.033406004387484425*Sinc[2*x]*
 Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.005744527287594969*
 Sinc[2*(-Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
(0.0033985547481202812 - 1.380366211082677*^-7*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
(0.0033957344119697245 + 1.3803662110828634*^-7*I)*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.005744527287594969*
 Sinc[2*(Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 8.15847931801218*^-9*Sinc[2*x]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 1.930922935277831*^-8*
 Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 1.0149358426852221*^-6*
 Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 1.0236118311995716*^-6*
 Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 1.930922935277831*^-8*
 Sinc[2*(Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.03951063403621358*Sinc[2*x]*
```

```
 Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.00679429066966187*
 Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.004018958617977918*
 Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.004016923980558435*
 Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.00679429066966187*
 Sinc[2*(Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.03340600438748425*Sinc[2*x]*
 Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.005744527287594969*
 Sinc[2*(-Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
(0.0033985547481202812 - 1.3803662110828634*^-7*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
(0.0033957344119697245 + 1.3803662110828634*^-7*I)*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.005744527287594969*
 Sinc[2*(Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.19426493431552047*Sinc[2*x]*
 Sinc[2*y]*Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
0.03340600104954632*Sinc[2*(-Pi + x)]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
(0.01975455832268188 + 1.3803662110765446*^-7*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*y]*Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
(0.019756074331108424 - 1.3803662110369542*^-7*I)*Sinc[2*(Pi/2 + x)]*
 Sinc[2*y]*Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
0.03340600104954632*Sinc[2*(Pi + x)]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.03340600438748425*Sinc[2*x]*
 Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.005744527287594969*
 Sinc[2*(-Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
(0.0033957344119697245 + 1.3803662110828634*^-7*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
(0.0033985547481202812 - 1.3803662110828634*^-7*I)*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.005744527287594969*
 Sinc[2*(Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.039510634036213585*Sinc[2*x]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.006794290669661873*
 Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.004016923980558435*
 Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.004018958617977918*
 Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.006794290669661873*
 Sinc[2*(Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 8.15847931911654*^-9*Sinc[2*x]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 1.9309229351673953*^-8*
 Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 1.0236118311995716*^-6*
 Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 1.0149358426852221*^-6*
 Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 1.9309229351673953*^-8*
 Sinc[2*(Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.03340600438748425*Sinc[2*x]*
 Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.005744527287594969*
 Sinc[2*(-Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
(0.0033957344119697245 + 1.380366211082677*^-7*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
(0.0033985547481202812 - 1.3803662110828634*^-7*I)*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.005744527287594969*
 Sinc[2*(Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 4.244042763161842*^-6*
 Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 4.2440427631618415*^-6*
 Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 6.9091114514986625*^-6*
 Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 6.9091114514986625*^-6*
 Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.033406003792346574*Sinc[2*x]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.005744537441177679*
 Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
(0.0033940849529615577 - 1.380366211067981*^-7*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
```

```
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
(0.003400207226272135 + 1.380366211062865*^-7*I)*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.005744537441177679*
 Sinc[2*(Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.033406003792346574*Sinc[2*x]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.005744537441177679*
 Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
(0.003400207226272135 + 1.380366211067981*^-7*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
(0.00339408495296155 77 - 1.380366211062865*^-7*I)*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.005744537441177679*
 Sinc[2*(Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 6.9091114514986625*^-6*
 Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 6.9091114514986625*^-6*
 Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 4.379697833326132*^-7*
 Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
4.379697833326133*^-7*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
1.0805575723769594*^-6*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
1.0805575723769594*^-6*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.1942648448105495*Sinc[2*x]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.03340597616795204*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
(0.01975467081815901 - 3.5144911800321895*^-8*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
(0.019755897494740927 + 3.5144911800144866*^-8*I)*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.03340597616795204*Sinc[2*(Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.19426484481054948*Sinc[2*x]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.03340597616795204*Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
(0.019755897494740927 + 3.5144911800321895*^-8*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*z]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + (0.01975467081815901 -
  3.5144911800144866*^-8*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*z]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.03340597616795204*Sinc[2*(Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
1.0805575723769594*^-6*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
1.0805575723769594*^-6*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
1.0805575723724215*^-6*Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
1.0805575723724215*^-6*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
1.7590992572972645*^-6*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + 1.7590992572972645*^-6*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(-Pi + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.033405975318093875*Sinc[2*x]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.005744526484050362*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - (0.0033963591397157628 -
  3.5144911803159567*^-8*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - (0.003397917905116128 +
  3.514491180424599*^-8*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + 0.005744526484050362*Sinc[2*(Pi + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.033405975318093875*Sinc[2*x]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.005744526484050362*Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + (0.003397917905116128 +
  3.514491180315988*^-8*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + (0.0033963591397157628 -
  3.5144911804245776*^-8*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
```

```
    Log[t/(1.*^-6 - t)])] - 0.0057445264840503 62*Sinc[2*(Pi + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
1.7590992572972645*^-6*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + 1.7590992572972645*^-6*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(Pi + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.1942648441300861*Sinc[2*x]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.033405974619761046*Sinc[2*(-Pi + x)]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
(0.019755476796745795 - 3.5144911802264 46*^-8*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
(0.019755090812208102 + 3.51449117982023*^-8*I)*Sinc[2*(Pi/2 + x)]*
 Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.033405974619761046*Sinc[2*(Pi + x)]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.0334059754696192*Sinc[2*x]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.0057445238988898745*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + (0.003397497174748252 -
  3.514491180334432*^-8*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + (0.003396779100852105 +
  3.514491180232661*^-8*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - 0.0057445238988898745*Sinc[2*(Pi + x)]*
 Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
2.0771956573394124*^-9*Sinc[2*x]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
4.916240124160756*^-9*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - 2.584085217645023*^-7*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
2.6061747855439034*^-7*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - 4.916240124160756*^-9*Sinc[2*(Pi + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.039510567960926 96*Sinc[2*x]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.006794276659946141*Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - 0.004018188761633151*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.004017670731205961*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + 0.006794276659946141*Sinc[2*(Pi + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.0334059754696192*Sinc[2*x]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.0057445238988898745*Sinc[2*(-Pi + x)]*Sinc[2*(Pi + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + (0.003397497174748252 -
  3.514491180334432*^-8*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + (0.003396779100852105 +
  3.514491180232661*^-8*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - 0.0057445238988898745*Sinc[2*(Pi + x)]*
 Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.1942648441300861*Sinc[2*x]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.033405974619761046*Sinc[2*(-Pi + x)]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
(0.019755090812208102 + 3.514491180226446*^-8*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*y]*Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - (0.019755476796745795 -
  3.514491179820 23*^-8*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + 0.033405974619761046*Sinc[2*(Pi + x)]*
 Sinc[2*y]*Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + 0.033405975469619195*Sinc[2*x]*
 Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.0057445238988898745*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - (0.003396779100852105 +
  3.514491180334432*^-8*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
```

```
    Log[t/(1.*^-6 - t)])] - (0.003397497174748252 -
  3.5144911802326 61*^-8*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + 0.00574452389888898745*Sinc[2*(Pi + x)]*
 Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.03951056796092695*Sinc[2*x]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.006794276659946141*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + 0.004017670731205961*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.004018188761633151*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - 0.006794276659946141*Sinc[2*(Pi + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
2.0771956573394124*^-9*Sinc[2*x]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
4.916240124160756*^-9*Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - 2.6061747855439034*^-7*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
2.584085217645023*^-7*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + 4.916240124160756*^-9*Sinc[2*(Pi + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.033405975469619195*Sinc[2*x]*Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.00574452389888898745*Sinc[2*(-Pi + x)]*Sinc[2*(Pi + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - (0.003396779100852105 +
  3.5144911803344 32*^-8*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - (0.003397497174748252 -
  3.5144911802326 61*^-8*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + 0.00574452389888898745*Sinc[2*(Pi + x)]*
 Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
1.0805575723724215*^-6*Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
1.0805575723724215*^-6*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
1.7590992572972645*^-6*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + 1.7590992572972645*^-6*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(-Pi + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.033405975318093875*Sinc[2*x]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.005744526484050362*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - (0.003396359139175 7628 -
  3.5144911803159 67*^-8*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - (0.003397917905116128 +
  3.5144911804245 99*^-8*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + 0.005744526484050362*Sinc[2*(Pi + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.033405975318093875*Sinc[2*x]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.005744526484050362*Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + (0.003397917905116128 +
  3.5144911803159 88*^-8*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + (0.003396359139175 7628 -
  3.5144911804245 776*^-8*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - 0.005744526484050362*Sinc[2*(Pi + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
1.7590992572972645*^-6*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*
 Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + 1.7590992572972645*^-6*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(Pi + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
2.423963549910466*^-6*Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
2.423963549910466*^-6*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
5.980396567743163*^-6*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
```

```
5.9803965677431 63*^-6*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.19426502871641924*Sinc[2*x]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.033406037070265064*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
(0.019751956480763583 - 1.9451118845113815*^-7*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
(0.019758745578803416 + 1.9451118845816005*^-7*I)*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.033406037070265064*Sinc[2*(Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.19426502871641926*Sinc[2*x]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.033406037070265064*Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
(0.019758745578803423 + 1.9451118843032146*^-7*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
(0.019751956480763 59 - 1.9451118843734337*^-7*I)*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.033406037070265064*Sinc[2*(Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
5.980396567743163*^-6*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
5.980396567743163*^-6*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
5.980396567742633*^-6*Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
5.980396567742634*^-6*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
9.73581817421927*^-6*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
9.735818174219 27*^-6*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.033406032366686655*Sinc[2*x]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.005744546691155196*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
(0.0033928398726404286 - 1.9451118843655309*^-7*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
(0.0034014669339341 84 + 1.9451118843631592*^-7*I)*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.005744546691155196*Sinc[2*(Pi + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.033406032366686655*Sinc[2*x]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.005744546691155198*Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
(0.0034014669339341 84 + 1.9451118844002253*^-7*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
(0.0033928398726404286 - 1.9451118843978536*^-7*I)*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.005744546691155198*Sinc[2*(Pi + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
9.735818174219 27*^-6*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
9.735818174219 27*^-6*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.19426502495036607*Sinc[2*x]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.033406028501733576*Sinc[2*(-Pi + x)]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
(0.019756417205623 33 - 1.9451118844262716*^-7*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
(0.019754280957926286 + 1.9451118843891546*^-7*I)*Sinc[2*(Pi/2 + x)]*
 Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.033406028501733576*Sinc[2*(Pi + x)]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.033406033205311 95*Sinc[2*x]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.005744532383467383*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] +
    Log[t/(1.*^-6 - t)])] + (0.003399138381585316 -
  1.9451118843989378*^-7*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] +
    Log[t/(1.*^-6 - t)])] + (0.0033951641706343465 +
  1.9451111884399141*^-7*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] +
    Log[t/(1.*^-6 - t)])] - 0.005744532383467383*Sinc[2*(Pi + x)]*
```

```
 Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
1.1496333628858867*^-8*Sinc[2*x]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
2.720916418911993*^-8*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] +
    Log[t/(1.*^-6 - t)])] - 1.430173326989669*^-6*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
1.4423988939827633*^-6*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] +
    Log[t/(1.*^-6 - t)])] - 2.720916418911993*^-8*Sinc[2*(Pi + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.039510700111558344*Sinc[2*x]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.006794304679397165*Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] +
    Log[t/(1.*^-6 - t)])] - 0.004019386383226745*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.004016519321051555*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] +
    Log[t/(1.*^-6 - t)])] + 0.006794304679397165*Sinc[2*(Pi + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.033406032320531195*Sinc[2*x]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.005744532383467383*Sinc[2*(-Pi + x)]*Sinc[2*(Pi + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] +
    Log[t/(1.*^-6 - t)])] + (0.003399138381585316 -
  1.9451118843989378*^-7*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] +
    Log[t/(1.*^-6 - t)])] + (0.0033951641706343465 +
  1.945111884399141*^-7*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] +
    Log[t/(1.*^-6 - t)])] - 0.005744532383467383*Sinc[2*(Pi + x)]*
 Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.19426502495036607*Sinc[2*x]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.033406028501733576*Sinc[2*(-Pi + x)]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
(0.019754280957926286 + 1.9451118844262843*^-7*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
(0.01975641720562333 - 1.9451111884389142*^-7*I)*Sinc[2*(Pi/2 + x)]*
 Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.033406028501733576*Sinc[2*(Pi + x)]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.03340603320531195*Sinc[2*x]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.005744532383467384*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
(0.0033951641706343474 + 1.9451111843989548*^-7*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
(0.003399138381585317 - 1.9451111843991242*^-7*I)*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.005744532383467384*Sinc[2*(Pi + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.039510700111558344*Sinc[2*x]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.006794304679397165*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.004016519321051555*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.004019386383226745*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.006794304679397165*Sinc[2*(Pi + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
1.1496333627754508*^-8*Sinc[2*x]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
2.720916419022429*^-8*Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
1.4423988939827633*^-6*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
1.430173326989669*^-6*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
2.720916419022429*^-8*Sinc[2*(Pi + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.03340603320531195*Sinc[2*x]*Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.005744532383467384*Sinc[2*(-Pi + x)]*Sinc[2*(Pi + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
(0.0033951641706343474 + 1.9451111843989548*^-7*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
```

```
     Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
    (0.003399138381585317 - 1.9451118843991242*^-7*I)*Sinc[2*(Pi/2 + x)]*
     Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
     Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
    0.005744532383467384*Sinc[2*(Pi + x)]*Sinc[2*(Pi + y)]*
     Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
    5.980396567742633*^-6*Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*(Pi + z)]*
     Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
    5.980396567742634*^-6*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*(Pi + z)]*
     Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
    9.735818174219277*^-6*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*
     Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
    9.735818174219277*^-6*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*
     Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
    0.033406032366686655*Sinc[2*x]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
     Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
    0.005744546691155196*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*
     Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
    (0.0033928398726404286 - 1.9451118843655309*^-7*I)*Sinc[2*(-Pi/2 + x)]*
     Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
     Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
    (0.003401466933934184 + 1.9451118843631592*^-7*I)*Sinc[2*(Pi/2 + x)]*
     Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
     Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
    0.005744546691155196*Sinc[2*(Pi + x)]*Sinc[2*(-Pi/2 + y)]*
     Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
    0.033406032366686655*Sinc[2*x]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
     Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
    0.005744546691155198*Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*
     Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
    (0.003401466933934184 + 1.9451118844002253*^-7*I)*Sinc[2*(-Pi/2 + x)]*
     Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
     Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
    (0.0033928398726404286 - 1.9451118843978536*^-7*I)*Sinc[2*(Pi/2 + x)]*
     Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
     Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
    0.005744546691155198*Sinc[2*(Pi + x)]*Sinc[2*(Pi/2 + y)]*
     Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
    9.735818174219277*^-6*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*
     Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
    9.735818174219277*^-6*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*
     Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])]],
uh2 -> Function[{x, y, z, t}, -0.19426493431552616*Sinc[2*(-Pi/2 + x)]*
     Sinc[2*y]*Sinc[2*z]*Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
    0.19426493431552616*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*z]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.033406001049540626*
     Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*z]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.033406001049540626*
     Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*z]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 1.720188108581885*^-6*Sinc[2*x]*
     Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
    4.244042776431004*^-6*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
    (0.01975607433110864 - 1.38036621092913 51*^-7*I)*Sinc[2*(-Pi/2 + x)]*
     Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
    (0.019754558322681696 + 1.3803662108407875*^-7*I)*Sinc[2*(Pi/2 + x)]*
     Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
    4.244042776431004*^-6*Sinc[2*(Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 1.7201881085810272*^-6*Sinc[2*x]*
     Sinc[2*(Pi/2 + y)]*Sinc[2*z]*Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
    4.244042776431861*^-6*Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
    (0.01975455832268169 + 1.380366210947471*^-7*I)*Sinc[2*(-Pi/2 + x)]*
     Sinc[2*(Pi/2 + y)]*Sinc[2*z]*Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
    (0.019756074331108636 - 1.3803662108591213*^-7*I)*Sinc[2*(Pi/2 + x)]*
     Sinc[2*(Pi/2 + y)]*Sinc[2*z]*Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
    4.244042776431861*^-6*Sinc[2*(Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.033406001049540626*
     Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*z]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.033406001049540626*
     Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*z]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.033406004387847674*
     Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*(-Pi + z)]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.033406004387847675*
     Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*(-Pi + z)]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.005744527287602 46*
     Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi + z)]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.005744527287602 46*
     Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi + z)]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 4.244042755299873*^-6*Sinc[2*x]*
     Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 6.909111431333901*^-6*
     Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
    (0.00339855474810932 - 1.380366214685622*^-7*I)*Sinc[2*(-Pi/2 + x)]*
     Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
    (0.0033957344119806727 + 1.3803662146856199*^-7*I)*Sinc[2*(Pi/2 + x)]*
     Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
```

```
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 6.909111431333901*^-6*
 Sinc[2*(Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 4.2440427552996435*^-6*Sinc[2*x]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 6.909111431334132*^-6*
 Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
(0.0033957344119806727 + 1.3803662147039943*^-7*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
(0.00339855474810932 - 1.38036621470394 51*^-7*I)*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 6.909111431334132*^-6*
 Sinc[2*(Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.00574452728760246*
 Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.00574452728760246*
 Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.1942649369881408*Sinc[2*x]*
 Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
0.03340600379234614*Sinc[2*(-Pi + x)]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 8.158474555652884*^-9*
 Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.03951063403621705*
 Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.03340600379234614*
 Sinc[2*(Pi + x)]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.03340600713028473*Sinc[2*x]*
 Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.005744537441173167*
 Sinc[2*(-Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 1.930923411528479*^-8*
 Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.006794290669658324*
 Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.005744537441173167*
 Sinc[2*(Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
(0.019752908736521198 - 1.3803662147040907*^-7*I)*Sinc[2*x]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
(0.00339408495296090 06 - 1.380366212999725*^-7*I)*Sinc[2*(-Pi + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 1.0149358463480349*^-6*
 Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.004016923980559744*
 Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
(0.00339408495296090 06 - 1.380366212999725*^-7*I)*Sinc[2*(Pi + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
(0.019757726682116474 + 1.3803662120995674*^-7*I)*Sinc[2*x]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
(0.003400207226271937 + 1.3803662155950352*^-7*I)*Sinc[2*(-Pi + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 1.0236118348967184*^-6*
 Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.004018958617976579*
 Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
(0.003400207226271937 + 1.3803662155950352*^-7*I)*Sinc[2*(Pi + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.03340600713028473*Sinc[2*x]*
 Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.005744537441173167*
 Sinc[2*(-Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 1.930923411528479*^-8*
 Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.006794290669658324*
 Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.005744537441173167*
 Sinc[2*(Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.1942649369881408*Sinc[2*x]*
 Sinc[2*y]*Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
0.03340600379234614*Sinc[2*(-Pi + x)]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.03951063403621705*
 Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 8.158474555652884*^-9*
 Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.03340600379234614*
 Sinc[2*(Pi + x)]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.03340600713028473*Sinc[2*x]*
 Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.005744537441173166*
 Sinc[2*(-Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.006794290669658324*
 Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 1.930923411528479*^-8*
```

```
 Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.005744537441173166*
 Sinc[2*(Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
(0.019757726682116478 + 1.380366212105206*^-7*I)*Sinc[2*x]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
(0.00340020272262719367 + 1.3803662156006738*^-7*I)*Sinc[2*(-Pi + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.0040189586179765775*
 Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 1.0236118348988868*^-6*
 Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
(0.00340020272262719367 + 1.3803662156006738*^-7*I)*Sinc[2*(Pi + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
(0.019752908736521198 - 1.380366214709194*^-7*I)*Sinc[2*x]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
(0.00339408495296089 98 - 1.380366212983 14*^-7*I)*Sinc[2*(-Pi + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.004016923980559743*
 Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 1.0149358463471675*^-6*
 Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
(0.00339408495296089 98 - 1.380366212983 14*^-7*I)*Sinc[2*(Pi + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.03340600713028473*Sinc[2*x]*
 Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.005744537441173166*
 Sinc[2*(-Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.006794290669658324*
 Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 1.930923411528479*^-8*
 Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.005744537441173166*
 Sinc[2*(Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.03340600438747674*
 Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.03340600438747675*
 Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.005744527287602 46*
 Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.005744527287602 46*
 Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 4.2440427552998 73*^-6*Sinc[2*x]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 6.9091114313339 01*^-6*
 Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
(0.00339855474810932 - 1.380366214685622*^-7*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
(0.00339573441198 06727 + 1.380366214685619 9*^-7*I)*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 6.9091114313339 01*^-6*
 Sinc[2*(Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 4.2440427552996 435*^-6*Sinc[2*x]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 6.9091114313341 32*^-6*
 Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
(0.00339573441198 06727 + 1.380366214703943*^-7*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
(0.00339855474810932 - 1.380366214703945 1*^-7*I)*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 6.9091114313341 32*^-6*
 Sinc[2*(Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.005744527287602 46*
 Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.005744527287602 46*
 Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.19426484413008688*
 Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.19426484413008688*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.033405974619760 26*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.033405974619760 25*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
4.3796978362002304*^-7*Sinc[2*x]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
1.080557573874633*^-6*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
(0.01975547679674573 - 3.514491180362547*^-8*I)*Sinc[2*(-Pi/2 + x)]*
```

```
 Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
(0.0197550908122082 + 3.5144911803625417*^-8*I)*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
1.080557573874633*^-6*Sinc[2*(Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
4.379697836287438*^-7*Sinc[2*x]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
1.080557573865912*^-6*Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
(0.019755090812208193 + 3.5144911802725186*^-8*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*z]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - (0.019755476796745726 -
  3.5144911802725246*^-8*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*z]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
1.080557573865912*^-6*Sinc[2*(Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.033405974619760 26*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.033405974619760 25*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.033405975469618 58*Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.033405975469618 196*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.005744523898890889*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + 0.005744523898890889*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(-Pi + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
1.080557571144335*^-6*Sinc[2*x]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
1.75909925526588 55*^-6*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + (0.0033974971747466687 -
  3.5144911854736 64*^-8*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - (0.003396779100853683 +
  3.5144911853632164*^-8*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - 1.7590992552658855*^-6*Sinc[2*(Pi + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
1.0805575711450157*^-6*Sinc[2*x]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
1.7590992552652045*^-6*Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + (0.0033967791008536826 +
  3.5144911853836 33*^-8*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - (0.003397497174746668 -
  3.5144911852732 12*^-8*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + 1.7590992552652045*^-6*Sinc[2*(Pi + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.005744523898890889*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + 0.005744523898890889*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(Pi + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.19426484481055017*Sinc[2*x]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.033405975318093833*Sinc[2*(-Pi + x)]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
2.0771950132170147*^-9*Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.039510567960927 42*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.033405975318093833*Sinc[2*(Pi + x)]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.033405976167952066*Sinc[2*x]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.005744526484049744*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - 4.916240764311908*^-9*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.006794276659945634*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + 0.005744526484049744*Sinc[2*(Pi + x)]*
 Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
(0.019754670818158723 - 3.5144911836527286*^-8*I)*Sinc[2*x]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
(0.003396359139175697 - 3.5144911822215 82*^-8*I)*Sinc[2*(-Pi + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
```

```
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
2.584085221270932*^-7*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + 0.0040176707312062 52*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
(0.003396359139175697 - 3.5144911822215 82*^-8*I)*Sinc[2*(Pi + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
(0.019755897494741374 + 3.5144911802138655*^-8*I)*Sinc[2*x]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
(0.0033979179051160926 + 3.5144911856348764*^-8*I)*Sinc[2*(-Pi + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
2.6061747894316756*^-7*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + 0.0040181887616328 4*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
(0.0033979179051160926 + 3.5144911856348764*^-8*I)*Sinc[2*(Pi + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.033405976167952066*Sinc[2*x]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.005744526484049744*Sinc[2*(-Pi + x)]*Sinc[2*(Pi + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - 4.916240764311908*^-9*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.006794276659945634*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + 0.005744526484049744*Sinc[2*(Pi + x)]*
 Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.1942648448105502*Sinc[2*x]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.033405975318093806*Sinc[2*(-Pi + x)]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.039510567960927 42*Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
2.07719501321701 47*^-9*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.033405975318093806*Sinc[2*(Pi + x)]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.033405976167952066*Sinc[2*x]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.005744526484049746*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + 0.006794276659945633*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
4.9162407651792694*^-9*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - 0.005744526484049746*Sinc[2*(Pi + x)]*
 Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
(0.019755897494741374 + 3.514491179999972*^-8*I)*Sinc[2*x]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
(0.003397917905116092 + 3.5144911858980317*^-8*I)*Sinc[2*(-Pi + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.004018188761632837*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - 2.60617478945335 96*^-7*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
(0.003397917905116092 + 3.5144911858980317*^-8*I)*Sinc[2*(Pi + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
(0.019754670818158723 - 3.514491183434951*^-8*I)*Sinc[2*x]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
(0.003396359139175697 7 - 3.514491182394117*^-8*I)*Sinc[2*(-Pi + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.004017670731206255*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + 2.584085221296953*^-7*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
(0.003396359139175697 7 - 3.514491182394117*^-8*I)*Sinc[2*(Pi + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.033405976167952066*Sinc[2*x]*Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.005744526484049746*Sinc[2*(-Pi + x)]*Sinc[2*(Pi + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
```

```
    Log[t/(1.*^-6 - t)])] + 0.006794276659945633*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
4.9162407651792694*^-9*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - 0.005744526484049746*Sinc[2*(Pi + x)]*
 Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.03340597546961818*Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.03340597546961896*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.005744523898890889*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + 0.005744523898890889*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(-Pi + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
1.0805575711144335*^-6*Sinc[2*x]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
1.7590992552658855*^-6*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + (0.0033974971747466687 -
  3.514491185473664*^-8*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - (0.003396779100853683 +
  3.5144911853632164*^-8*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - 1.7590992552658855*^-6*Sinc[2*(Pi + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
1.0805575711450157*^-6*Sinc[2*x]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
1.7590992552652045*^-6*Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + (0.0033967791008536826 +
  3.514491185383633*^-8*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - (0.003397497174746668 -
  3.5144911185273212*^-8*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + 1.7590992552652045*^-6*Sinc[2*(Pi + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.005744523898890889*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*
 Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + 0.005744523898890889*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(Pi + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.1942650249503809*Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.1942650249503809*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.03340602850171875*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.03340602850171754*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
2.4239635644669265*^-6*Sinc[2*x]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
5.980396613477894*^-6*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
(0.019756417205627885 - 1.9451118821782975*^-7*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
(0.01975428095792176 + 1.9451118821782964*^-7*I)*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
5.980396613477894*^-6*Sinc[2*(Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
2.4239635644669265*^-6*Sinc[2*x]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
5.980396613478002*^-6*Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
(0.019754280957921755 + 1.9451118821979362*^-7*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
(0.019756417205627878 - 1.9451118821979373*^-7*I)*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
5.980396613478002*^-6*Sinc[2*(Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.03340602850171875*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.03340602850171754*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.033406033205292245*Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.033406033205292245*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.005744532383486795*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*
```

```
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.005744532383486795*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
5.98039655331690 56*^-6*Sinc[2*x]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
9.735818095504493*^-6*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
(0.003995138381560354 - 1.9451118903611 27*^-7*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
(0.003395164170659298 + 1.9451118903500796*^-7*I)*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
9.735818095504493*^-6*Sinc[2*(Pi + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
5.980396553317465*^-6*Sinc[2*x]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
9.735818095504373*^-6*Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
(0.003395164170659297 + 1.9451118903807629*^-7*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
(0.0033991383815603535 - 1.9451118903697218*^-7*I)*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
9.735818095504373*^-6*Sinc[2*(Pi + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.005744532383486795*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.005744532383486795*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.19426502871643148*Sinc[2*x]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.033406032366685524*Sinc[2*(-Pi + x)]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
1.1496321256077024*^-8*Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.03951070011156743*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.033406032366685524*Sinc[2*(Pi + x)]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.033406037070265654*Sinc[2*x]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.005744546691143478*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] +
    Log[t/(1.*^-6 - t)])] - 2.720917658804306*^-8*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.006794304679387975*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] +
    Log[t/(1.*^-6 - t)])] + 0.005744546691143478*Sinc[2*(Pi + x)]*
 Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
(0.019751956480745188 - 1.9451119014881372*^-7*I)*Sinc[2*x]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
(0.00339283987263810 75 - 1.945111890836935*^-7*I)*Sinc[2*(-Pi + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
1.43017334111421 31*^-6*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] +
    Log[t/(1.*^-6 - t)])] + 0.004016519321050392*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
(0.00339283987263810 75 - 1.945111890836935*^-7*I)*Sinc[2*(Pi + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
(0.01975874557882411 + 1.9451118947299096*^-7*I)*Sinc[2*x]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
(0.003401466933934258 + 1.9451118976268978*^-7*I)*Sinc[2*(-Pi + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
1.4423989081586118*^-6*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] +
    Log[t/(1.*^-6 - t)])] + 0.004019386383227855*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
(0.003401466933934258 + 1.9451118976268978*^-7*I)*Sinc[2*(Pi + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.033406037070265654*Sinc[2*x]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.005744546691143478*Sinc[2*(-Pi + x)]*Sinc[2*(Pi + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] +
    Log[t/(1.*^-6 - t)])] - 2.720917658804306*^-8*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.006794304679387975*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*
```

```
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] +
    Log[t/(1.*^-6 - t)])] + 0.005744546691143478*Sinc[2*(Pi + x)]*
 Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.19426502871643148*Sinc[2*x]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.03340603236668552*Sinc[2*(-Pi + x)]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.039510700111567425*Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
1.1496321249138l3*^-8*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.03340603236668552*Sinc[2*(Pi + x)]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.033406037070265654*Sinc[2*x]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.005744546691143481*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.006794304679387974*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
2.7209176589777784*^-8*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.005744546691143481*Sinc[2*(Pi + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
(0.019758745578824108 + 1.9451118947137202*^-7*I)*Sinc[2*x]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
(0.003401466933934257 + 1.9451118976107084*^-7*I)*Sinc[2*(-Pi + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.004019386382278555*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
1.4423989081599128*^-6*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
(0.003401466933934257 + 1.9451118976107084*^-7*I)*Sinc[2*(Pi + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
(0.019751956480745l9 - 1.9451119015145247*^-7*I)*Sinc[2*x]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
(0.003392839872638108 - 1.9451118908416385*^-7*I)*Sinc[2*(-Pi + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.004016519321050392l*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
1.4301733411163815*^-6*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
(0.003392839872638108 - 1.9451118908416385*^-7*I)*Sinc[2*(Pi + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.033406037070265654*Sinc[2*x]*Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.005744546691143481*Sinc[2*(-Pi + x)]*Sinc[2*(Pi + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.006794304679387974*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
2.7209176589777784*^-8*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.005744546691143481*Sinc[2*(Pi + x)]*Sinc[2*(Pi + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.03340603320529245*Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.03340603320529245*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.005744532383486795*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.005744532383486795*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
5.9803965533169056*^-6*Sinc[2*x]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
9.735818095504493*^-6*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
(0.003399138381560354 - 1.9451118903611l7*^-7*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
(0.003395164170659298 + 1.9451118903500796*^-7*I)*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
9.735818095504493*^-6*Sinc[2*(Pi + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
5.980396553317465*^-6*Sinc[2*x]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
9.735818095504373*^-6*Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
(0.003395164170659297 + 1.9451118903807629*^-7*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
(0.003399138381560353 5 - 1.9451118903697218*^-7*I)*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
```

```
     Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
    9.7358180955043 73*^-6*Sinc[2*(Pi + x)]*Sinc[2*(Pi/2 + y)]*
     Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
    0.0057445323834868795*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*
     Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
    0.0057445323834868795*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*
     Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])]],
uh3 -> Function[{x, y, z, t}, 0.19426493698814232*Sinc[2*(-Pi/2 + x)]*
     Sinc[2*y]*Sinc[2*z]*Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
    0.19426493698814232*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*z]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.03340600379234267*
     Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*z]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.03340600379234266*
     Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*z]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.19426493431553335*Sinc[2*x]*
     Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
    0.03340600438746527*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 8.1584750899477 14*^-9*
     Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.039510634036218836*
     Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.03340600438746527*
     Sinc[2*(Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.19426493431553335*Sinc[2*x]*
     Sinc[2*(Pi/2 + y)]*Sinc[2*z]*Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
    0.033406004387465264*Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.039510634036218815*
     Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 8.158475096886608*^-9*
     Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.033406004387465264*
     Sinc[2*(Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.03340600379234267*
     Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*z]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.03340600379234266*
     Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*z]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.033406007130283175*
     Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*(-Pi + z)]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.03340600713028319*
     Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*(-Pi + z)]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.005744537441176652*
     Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi + z)]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.005744537441176652*
     Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi + z)]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.03340600104954768*Sinc[2*x]*
     Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.005744527287599651*
     Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 1.930923356624481*^-8*
     Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.006794290669 6565105*
     Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.005744527287599651*
     Sinc[2*(Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.03340600104954768*Sinc[2*x]*
     Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.005744527287599649*
     Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.006794290669 6565105*
     Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 1.9309233567112172*^-8*
     Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.005744527287599649*
     Sinc[2*(Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.005744537441176652*
     Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi + z)]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.005744537441176652*
     Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi + z)]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 1.720188110199284*^-6*Sinc[2*x]*
     Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
    4.244042768947355*^-6*Sinc[2*(-Pi + x)]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
    (0.019752908736516053 - 1.3803662131716263*^-7*I)*Sinc[2*(-Pi/2 + x)]*
     Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
    (0.019757726682121578 + 1.3803662105719 91*^-7*I)*Sinc[2*(Pi/2 + x)]*
     Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
    4.244042768947355*^-6*Sinc[2*(Pi + x)]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 4.244042766820049*^-6*Sinc[2*x]*
     Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 6.909111410747895*^-6*
     Sinc[2*(-Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
    (0.003394084952967016 - 1.3803662076233 51*^-7*I)*Sinc[2*(-Pi/2 + x)]*
     Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
    (0.00340020722626581 + 1.3803662102229884*^-7*I)*Sinc[2*(Pi/2 + x)]*
     Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
     Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 6.909111410747895*^-6*
     Sinc[2*(Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
```

```
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
(0.01975607433111183 - 1.3803662150764983*^-7*I)*Sinc[2*x]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
(0.003398554748116338 - 1.3803662143479144*^-7*I)*Sinc[2*(-Pi + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 1.0149358428647101*^-6*
 Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.004018958617986247*
 Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
(0.003398554748116338 - 1.3803662143479144*^-7*I)*Sinc[2*(Pi + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
(0.019754558322682452 + 1.3803662176713295*^-7*I)*Sinc[2*x]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
(0.003395734411969688 + 1.3803662117515856*^-7*I)*Sinc[2*(-Pi + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.004016923980550068*
 Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 1.023611831355714*^-6*
 Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
(0.003395734411969688 + 1.3803662117515856*^-7*I)*Sinc[2*(Pi + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 4.244042766820049*^-6*Sinc[2*x]*
 Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 6.9091111410747895*^-6*
 Sinc[2*(-Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
(0.003394084952967016 - 1.3803662076233351*^-7*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
(0.00340020722626581 + 1.3803662102229884*^-7*I)*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 6.9091111410747895*^-6*
 Sinc[2*(Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 1.7201881101932226*^-6*Sinc[2*x]*
 Sinc[2*y]*Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
4.244042768948524*^-6*Sinc[2*(-Pi + x)]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
(0.01975772668212158 + 1.3803662105692666*^-7*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*y]*Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
(0.019752908736516056 - 1.380366213168904*^-7*I)*Sinc[2*(Pi/2 + x)]*
 Sinc[2*y]*Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
4.244042768948524*^-6*Sinc[2*(Pi + x)]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 4.244042766823742*^-6*Sinc[2*x]*
 Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 6.9091111410490915*^-6*
 Sinc[2*(-Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
(0.0034002072262658144 + 1.3803662102202652*^-7*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
(0.003394084952967017 - 1.3803662076206258*^-7*I)*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 6.9091111410490915*^-6*
 Sinc[2*(Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
(0.019754558322682456 + 1.3803662176701240*^-7*I)*Sinc[2*x]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
(0.003395734411969685 + 1.3803662117503802*^-7*I)*Sinc[2*(-Pi + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 1.023611831351811*^-6*
 Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.004016923980550063*
 Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
(0.003395734411969685 + 1.3803662117503802*^-7*I)*Sinc[2*(Pi + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
(0.019756074331111183 - 1.3803662150737952*^-7*I)*Sinc[2*x]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
(0.00339855474811634 - 1.3803662143495482*^-7*I)*Sinc[2*(-Pi + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.004018958617986247*
 Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 1.0149358428625417*^-6*
 Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
(0.00339855474811634 - 1.3803662143495482*^-7*I)*Sinc[2*(Pi + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 4.244042766823742*^-6*Sinc[2*x]*
 Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 6.9091111410490915*^-6*
 Sinc[2*(-Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
```

```
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
(0.0034002072262658144 + 1.3803662102202652*^-7*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
(0.003394084952967017 - 1.3803662076206258*^-7*I)*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 6.9091114107490915*^-6*
 Sinc[2*(Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.033406007130283175*
 Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.03340600713028319*
 Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.005744537441176652*
 Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.005744537441176652*
 Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.03340600104954768*Sinc[2*x]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.005744527287599651*
 Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 1.930923356624481*^-8*
 Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.006794290669656510 5*
 Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.005744527287599651*
 Sinc[2*(Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.03340600104954768*Sinc[2*x]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.005744527287599649*
 Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.006794290669656510 5*
 Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 1.9309233567112172*^-8*
 Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.005744527287599649*
 Sinc[2*(Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.005744537441176652*
 Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] - 0.005744537441176652*
 Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] + 0.19426484481055037*
 Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.19426484481055037*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.033405975031809336*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.033405975031809335*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.19426484413008782*Sinc[2*x]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.033405975469616614*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
2.077195068728166*^-9*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.03951056796092764*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.033405975469616614*Sinc[2*(Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.19426484481300878*Sinc[2*x]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.033405975469616635*Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.03951056796092764*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
2.07719507566706*^-9*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.033405975469616635*Sinc[2*(Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.033405975031809336*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.033405975031809335*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.033405976167951 85*Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.033405976167951 85*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.005744526484050235 5*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - 0.005744526484050235 5*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(-Pi + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.033405974619761 2*Sinc[2*x]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.005744523898890514*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - 4.916240696657692*^-9*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
```

```
0.0067942766599453875*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - 0.005744523898890514*Sinc[2*(Pi + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.0334059746197612*Sinc[2*x]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.005744523898890513*Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - 0.0067942766599453875*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
4.916240696657692*^-9*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + 0.005744523898890513*Sinc[2*(Pi + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.005744526484050235 5*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - 0.005744526484050235 5*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(Pi + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
4.379697836606021*^-7*Sinc[2*x]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
1.0805557572602192*^-6*Sinc[2*(-Pi + x)]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
(0.019754670818158175 - 3.5144911818381826*^-8*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
(0.019755897494741888 + 3.51449117830917*^-8*I)*Sinc[2*(Pi/2 + x)]*
 Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
1.0805557572602192*^-6*Sinc[2*(Pi + x)]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
1.080557572377735*^-6*Sinc[2*x]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
1.7590992531286808*^-6*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - (0.003396359139176669 -
  3.5144911762108944*^-8*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + (0.003397917905115116 +
  3.514491179739 84*^-8*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - 1.7590992531286808*^-6*Sinc[2*(Pi + x)]*
 Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
(0.019755476796746108 - 3.5144911840904905*^-8*I)*Sinc[2*x]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
(0.00339749717474772 - 3.514491184827748*^-8*I)*Sinc[2*(-Pi + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
2.5840852166392203*^-7*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - 0.004018188761634144 4*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
(0.00339749717474772 - 3.514491184827748*^-8*I)*Sinc[2*(Pi + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
(0.019755090812208 35 + 3.5144911875725066*^-8*I)*Sinc[2*x]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
(0.003396779100852104 + 3.514491181327502*^-8*I)*Sinc[2*(-Pi + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.004017670731204946 5*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - 2.60617478438363*^-7*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
(0.003396779100852104 + 3.514491181327502*^-8*I)*Sinc[2*(Pi + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
1.0805557572377735*^-6*Sinc[2*x]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
1.7590992531286808*^-6*Sinc[2*(-Pi + x)]*Sinc[2*(Pi + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - (0.003396359139176669 -
  3.5144911762108944*^-8*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + (0.003397917905115116 +
  3.514491179739 84*^-8*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - 1.7590992531286808*^-6*Sinc[2*(Pi + x)]*
 Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
4.3796978366198045*^-7*Sinc[2*x]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
```

```
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
1.0805575726026522*^-6*Sinc[2*(-Pi + x)]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
(0.01975589749474189 + 3.514491178299842*^-8*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*y]*Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + (0.01975467081815817 -
  3.514491181828798*^-8*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + 1.0805575726026522*^-6*Sinc[2*(Pi + x)]*
 Sinc[2*y]*Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + 1.0805575723771528*^-6*Sinc[2*x]*
 Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
1.7590992531274244*^-6*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - (0.0033979179051151155 +
  3.514491179730511*^-8*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + (0.0033963591391766696 -
  3.514491176201565*^-8*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + 1.7590992531274244*^-6*Sinc[2*(Pi + x)]*
 Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
(0.019755090812220835 + 3.514491187613718*^-8*I)*Sinc[2*x]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
(0.003396779100852104 + 3.5144911812819774*^-8*I)*Sinc[2*(-Pi + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
2.606174784387967*^-7*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - 0.0040176707312049465*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
(0.003396779100852104 + 3.5144911812819774*^-8*I)*Sinc[2*(Pi + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
(0.019755476796746108 - 3.5144911848048425*^-8*I)*Sinc[2*x]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
(0.003397497174747721 - 3.5144911848724184*^-8*I)*Sinc[2*(-Pi + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.004018188761634144 4*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + 2.584085216647894*^-7*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
(0.003397497174747721 - 3.5144911848724184*^-8*I)*Sinc[2*(Pi + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
1.0805575723771528*^-6*Sinc[2*x]*Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
1.7590992531274244*^-6*Sinc[2*(-Pi + x)]*Sinc[2*(Pi + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - (0.0033979179051151155 +
  3.514491179730511*^-8*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + (0.0033963591391766696 -
  3.514491176201565*^-8*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + 1.7590992531274244*^-6*Sinc[2*(Pi + x)]*
 Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.03340597616795185*Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.03340597616795185*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.005744526484050235 5*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - 0.005744526484050235 5*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(-Pi + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.0334059746197612*Sinc[2*x]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
0.005744523898890514*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - 4.916240696657692*^-9*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.006794276659945387 5*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - 0.005744523898890514*Sinc[2*(Pi + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.0334059746197612*Sinc[2*x]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.005744523898890513*Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*
```

```
 Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - 0.0067942766599453875*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
4.916240696657692*^-9*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] + 0.005744523898890513*Sinc[2*(Pi + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.0057445264840502355*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*
 Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) +
    Log[t/(1.*^-6 - t)])] - 0.0057445264840502355*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(Pi + y)]*Sinc[2*(Pi + z)]*
 Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
0.19426502871643547*Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.19426502871643547*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.033406032366676504*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.033406032366676651*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.19426502495039955*Sinc[2*x]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.033406033205262265*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
1.1496322643855805*^-8*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.039510700111572095*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.033406033205262265*Sinc[2*(Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.19426502495039952*Sinc[2*x]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.033406033205262265*Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.039510700111572095*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
1.1496322650794699*^-8*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.033406033205262265*Sinc[2*(Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.033406032366676504*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.033406032366676651*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*z]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.033406037070262164*Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.033406037070262164*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.005744546691152506*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.005744546691152506*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.033406028501737715*Sinc[2*x]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.005744532383479583*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
2.7209175158630916*^-8*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.006794304679383288*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.005744532383479583*Sinc[2*(Pi + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.033406028501737715*Sinc[2*x]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.005744532383479583*Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.006794304679383288*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
2.7209175156896193*^-8*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.005744532383479583*Sinc[2*(Pi + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.005744546691152506*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.005744546691152506*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*
 Sinc[2*(-Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
2.4239635754677605*^-6*Sinc[2*x]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
5.980396603540217*^-6*Sinc[2*(-Pi + x)]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
(0.01975195648072578 7 - 1.9451118957165143*^-7*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
(0.019758745578843447 + 1.9451118889551617*^-7*I)*Sinc[2*(Pi/2 + x)]*
 Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
5.980396603540217*^-6*Sinc[2*(Pi + x)]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
```

```
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
5.980396595435747*^-6*Sinc[2*x]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
9.735818017593764*^-6*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] +
    Log[t/(1.*^-6 - t)])] - (0.003392839872648702 -
  1.945111876729911*^-7*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] +
    Log[t/(1.*^-6 - t)])] + (0.0034014669339236486 +
  1.945111883491252*^-7*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] +
    Log[t/(1.*^-6 - t)])] - 9.735818017593764*^-6*Sinc[2*(Pi + x)]*
 Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
(0.01975641720563834 - 1.9451119005861949*^-7*I)*Sinc[2*x]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
(0.003399138381574971 - 1.9451118922725326*^-7*I)*Sinc[2*(-Pi + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
1.4301733322497762*^-6*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] +
    Log[t/(1.*^-6 - t)])] - 0.00401938638325325*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
(0.003399138381574971 - 1.9451118922725326*^-7*I)*Sinc[2*(Pi + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
(0.019754280957921626 + 1.9451119073480568*^-7*I)*Sinc[2*x]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
(0.0033951641706343482 + 1.945111885512225*^-7*I)*Sinc[2*(-Pi + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
0.00401651932102497*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] +
    Log[t/(1.*^-6 - t)])] - 1.4423988992377963*^-6*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
(0.0033951641706343482 + 1.945111885512225*^-7*I)*Sinc[2*(Pi + x)]*
 Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
5.980396595435747*^-6*Sinc[2*x]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
9.735818017593764*^-6*Sinc[2*(-Pi + x)]*Sinc[2*(Pi + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] +
    Log[t/(1.*^-6 - t)])] - (0.003392839872648702 -
  1.945111876729911*^-7*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] +
    Log[t/(1.*^-6 - t)])] + (0.0034014669339236486 +
  1.945111883491252*^-7*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*
 Sinc[2*(-Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] +
    Log[t/(1.*^-6 - t)])] - 9.735818017593764*^-6*Sinc[2*(Pi + x)]*
 Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
2.4239635754637507*^-6*Sinc[2*x]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
5.980396603545911*^-6*Sinc[2*(-Pi + x)]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
(0.01975875457884345 + 1.9451118889547509*^-7*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
(0.01975195648072579 - 1.9451118957161024*^-7*I)*Sinc[2*(Pi/2 + x)]*
 Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
5.980396603545911*^-6*Sinc[2*(Pi + x)]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
5.980396595435075*^-6*Sinc[2*x]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
9.735818017592754*^-6*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
(0.0034014669339236486 + 1.9451118834908423*^-7*I)*Sinc[2*(-Pi/2 + x)]*
 Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
(0.0033928398726486998 - 1.945111876729495*^-7*I)*Sinc[2*(Pi/2 + x)]*
 Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
9.735818017592754*^-6*Sinc[2*(Pi + x)]*Sinc[2*(-Pi + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
(0.019754280957921626 + 1.9451119073467656*^-7*I)*Sinc[2*x]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
(0.0033951641706343482 + 1.9451118855109338*^-7*I)*Sinc[2*(-Pi + x)]*
 Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
 Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
1.442398899235628*^-6*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
0.004016519321024967 5*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*
 Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
```

```
      (0.003395164170634348 + 1.9451118855109338*^-7*I)*Sinc[2*(Pi + x)]*
       Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
       Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
      (0.019756417205638345 - 1.945111900586646*^-7*I)*Sinc[2*x]*
       Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
       Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
      (0.0033991383815749716 - 1.9451118922727978*^-7*I)*Sinc[2*(-Pi + x)]*
       Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
       Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
      0.0040193863832532 53*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*
       Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
      1.4301733322471 74*^-6*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*
       Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
      (0.0033991383815749716 - 1.9451118922727978*^-7*I)*Sinc[2*(Pi + x)]*
       Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
       Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
      5.980396595435075*^-6*Sinc[2*x]*Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
       Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
      9.735818017592754*^-6*Sinc[2*(-Pi + x)]*Sinc[2*(Pi + y)]*
       Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
      (0.003401466933923649 + 1.9451118834908423*^-7*I)*Sinc[2*(-Pi/2 + x)]*
       Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
       Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
      (0.0033928398726486998 - 1.945111876729495*^-7*I)*Sinc[2*(Pi/2 + x)]*
       Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
       Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
      9.735818017592754*^-6*Sinc[2*(Pi + x)]*Sinc[2*(Pi + y)]*
       Sinc[2*(Pi/2 + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
      0.033406037070261 64*Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*(Pi + z)]*
       Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
      0.033406037070261 64*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*(Pi + z)]*
       Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
      0.005744546691152506*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*
       Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
      0.005744546691152506*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*
       Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
      0.033406028501737 15*Sinc[2*x]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
       Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
      0.005744532383479583*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*
       Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
      2.7209175158630916*^-8*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*
       Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
      0.006794304679383288*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*
       Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
      0.005744532383479583*Sinc[2*(Pi + x)]*Sinc[2*(-Pi/2 + y)]*
       Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
      0.033406028501737 15*Sinc[2*x]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
       Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
      0.005744532383479583*Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*
       Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
      0.006794304679383288*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*
       Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
      2.7209175156896193*^-8*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*
       Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
      0.005744532383479583*Sinc[2*(Pi + x)]*Sinc[2*(Pi/2 + y)]*
       Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
      0.005744546691152506*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*
       Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
      0.005744546691152506*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*
       Sinc[2*(Pi + z)]*Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])]],
  ph -> Function[{x, y, z, t},
    (-0.3052182619061430215482109477056309657537937319304821472724386150590 79\
8139923366688593523233029389895839063837889426786351527853610672 4`120.1505149\
9783198 + 4.450360217818520156093777167065522646893889813389055260844866475 17\
0738876818764631040592614365185657489185188361541518402742 52`111.314299647691\
98*^-10*I)*Sinc[2*x]*Sinc[2*y]*Sinc[2*z]*Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (0.1298380446658653368429130879043204503724905835292113825437316365135 60\
5734769447730148263555638920927649820552726328340982319536200636 47`120.147660\
50687132 + 9.812372038269167508512509215962678282199129079143396454997285243\
025012824869102366735762008509221966978747993232509119642103 3`112.02603254660\
754*^-10*I)*Sinc[2*(-Pi + x)]*Sinc[2*y]*Sinc[2*z]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (0.4447115939453458326445125582454895859343475672009781401041680935445 62\
7280747748059246016065781137625475299111667379217017403991366356 16`120.150509\
5025504 + 2.221497399064069294318407680818560358825721828461585337425439494 10\
3314765205290178426960174808134319306316675294644445501314 53`110.849076859693\
88*^-10*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*z]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (0.4447115939453458681315655084836976266089430124596017617140102051465 22\
0902455052058539144381003555286777513790338693634347603255835247 4`120.1505095\
025504 + 2.221497680327678031950980046532600351743060485684625268206797692 187\
2763180079983283403530314215446743848763464598686577162408 1`110.84907691467987\
*^-10*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*z]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (0.1298380446658653260394718288547594544284429636217912989904956061802 86\
3888566405838844717965821272714649397960616398593035645555057270 9`120.147660\
50687132 + 9.812371667579138075432496475724705683139792126374735975016113 86\
1919191506103527177414430875881892703169224847310829071662725`112.02603253020\
082*^-10*I)*Sinc[2*(Pi + x)]*Sinc[2*y]*Sinc[2*z]*
```

```
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (0.12983803921825049986982890523834904009529142742390951789729423703309\
6934274139377040461087380430354749569052286381277780079603944133 3`120.147660\
6067214 - 1.934842146797602040910323997234665159448063141163264258033932148 01\
1895502725593995542980973715022795281763114090658844250123568 3`112.32090419754\
648*^-9*I)*Sinc[2*x]*Sinc[2*(-Pi + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (2.5149926480667000323113833955090591408528230310086350980240580798398789\
5767950011248702710239089399747016477782604661991248642576847697 27`120.129684\
58692983 - 2.42332194274226798457784299627811327853195623395770771900488523937\
9678342419606846784124306218750759961550824933746732594008844 6`112.1135589818\
6895*^-8*I)*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.1863454811659613074727305278316323217491640968040151907128804304643821\
6156936040357181529291554838535660714092647564162490615283497867 `119.6829061\
0225135 + 0.215337127715830506465474520084503064177320943356283787929316281 33\
7393005258942071821936933897829242054783170387355945907434050633577932 `119.74\
570615247134*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.2174601423346211389172064361347562086264080991190266031740221421688177\
1707029516609471544480539276684959754254162050914363770369492691 11`119.742274\
65398712 + 0.207122618474576923707663091785134390311417222363292216212850943 7\
4038202497212058323611852747099962092549235542669462662419459664214508 4`119.7\
2112251390776*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (2.6169589334966910952674762515603451182889564904527209641376080027750 79\
5790879685930345562936272337703358947290953764976612749872758144 5`120.128181\
67570661 + 2.431529965378705380457471280989762749095397827143317566414273474\
1275920425951832256617119748670807174518665456260681254518026 1`112.096264394\
19878*^-8*I)*Sinc[2*(Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (0.4447115851808933316689195646260232439079592617951282007830363646656762\
2400741267338279227954223667123423423328836253058591331041719657 13`120.150509\
50257431 + 5.315622520237123581615043553075768455144145434509676160841126 6711\
8949267536962442249309123154849559470147619946485551046242 59`111.227985191865\
36*^-10*I)*Sinc[2*x]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (2.3150240231497370464074830659119305134238188220770325209568005827957 78\
3654749715886481085705614082575170656143750555971195930937485071 44`120.143252\
05647128 + 0.215337128255901616343709862432424279303025144903721523407400865 8\
1226450991560448511547050957048190742003878308065332387070551941640080 8`119.1\
1181547138067*I)*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.1663543212095782337475551300194621414881548969621771169638992265205656\
3248331584812039846972574054500156338099606989918347585856554349 12`119.931437\
4536839 + 4.368522652012921632203473731767915673505376466850394934362646165945\
3472767038185896821294102160398153118582304148539234606882384 `112.35073795911\
796*^-9*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.1684799741442829155924084774871511349600542894991874869562501107643 41\
0467777563958060030756721259496636569023110598723213374430682520 49`119.958710\
01933277 - 3.631476321677396684822037302785622302801319989096851153850222 1791\
1651367906132222144317169669235612240443136505120731340415462 2`112.2922449489\
1564*^-9*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (2.3339439763530621833346223579454822716304755443221432104631496563450 56\
7711509845702397875789826656840388399449148664005020254250749168 23`120.144028\
6221726 + 0.207122620237098915136608935602700825102865460090354750126836839 05\
7398516773106526474181008854074873811902334151893398460627737822336072 `119.09\
216572665063*I)*Sinc[2*(Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (0.4447115851808933072780344299169914240195965790194195849223108932269 46\
5488428669612825190824593539894766270112548462936663769214420604 48`120.150509\
5025743 + 5.315622196482456335049270334573683931851788435068706265698868 41587\
9419951327287043845870499465059363888458056659359557773988 68`111.22798516541408\
*^-10*I)*Sinc[2*x]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (2.3339439763530623783062564941286112196199744964190632921583544055560 25\
5329947983724731646199871661684432836297176098006900415292907297 35`120.144028\
62217257 + 0.207122620237098967630864157016548833692657575053940273999624542 4\
6937613721679600957618611613452647000119008190026240937668836740972 38`119.0\
9216572665062*I)*Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.1684799741442829211609442481634457539620888420675807254571125311391528\
2464280996979251129717654399571317556391615040154055135076786966 01`119.958710\
01933277 - 3.631476309229412800947665408696513775396088396141201352966959693\
4752928780739467501214753953605155264597893594251094439601825 8`112.2922449474\
2694*^-9*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.1663543212095782273723676595009471312958247552423367994732646303518 14\
1892328537448871237792748590185404169243081973680086613253046089 21`119.931437\
4536839 + 4.368522640505234511105745591944782342990945466551361597237539333 0\
0539597254538444778974356112707418830205380368757237262113801 `112.35073795797\
395*^-9*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (2.3150240231497370329291979866136178416914668554722733229290631663940 09\
2585731624200716248554282750216562164584623518776582582091780408 47`120.143252\
05647128 + 0.215337128255901595963592268916859219127905441183132759284893359\
9248022449710875995849414207413521220745724283638169206074088270644789 7`119.1\
```

```
11815471380674*I)*Sinc[2*(Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (0.12983803921825050897901311249916638132145438510887875765715349053018 4\
3041568477535075336833075634655857924086726800662754022074889828 78`120.1476606\
0672143 - 1.934842181481414018142885165067732795289011073933086710046185908184\
67943850250801472797378096955028792374696416439288053311201188`112.3209042053316\
*^-9*I)*Sinc[2*x]*Sinc[2*(Pi + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (2.616958933496690926666911930818134904677178876072253940743729295018711\
0972811244454752284790729712469218940297339610446238964190331960 98`120.128181\
67570663 + 2.431529969058510163985851729136423294920032780139229834607309581 6\
4557451162819183691050888790775491284446273285722811816562506 4`112.096264394\
85608*^-8*I)*Sinc[2*(-Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.217460142334621398876134404929023706893479538080476253926287659118893\
3720605868873264244988374421767705633740934908103745898782404684 12`119.742274\
65398714 + 0.207122618474576916971111207695601322951666589956101673631763882 8\
0027686886233737111832537714486613202508286510101275376966691322650298 46`119.7\
2112251390777*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.186345481165961110921581598751287065836503396499987711144145988807018\
4682754764115018608053287807891095988514833218500443804580192153 22`119.682906\
10225135 + 0.215337127715830478693844419546926525278365328051545503007562550 5\
7995400356067533677810152238595594533396294205272796878664866404735778 89`119.7\
4570615247134*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (2.514992648066699287956489713930524148123081746725640763246185936494642\
5552635648990813148936181029776411465043472993451418292947190530 66`120.129684\
58692984 - 2.423321929776958646402297748017850306323966529464426413932744593 7\
6482344294129838741234262390644487201869630349869731615374957 13`112.113558979\
54537*^-8*I)*Sinc[2*(Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (0.129838050405470073472203648913607406282339746923343538217002068414968\
2313554984002305568542478766504052855812620445908390294159472872 21`120.147660\
58893775 - 1.207279253478690785678724290129993121962575116130123125550508644 8\
1542406013041544115827541701950122375439206082339870349127482`111.11606634080\
89*^-10*I)*Sinc[2*x]*Sinc[2*y]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (2.514992770602591213047642167502406166858292353509532220269790578601069\
3360699775258202057978399794352887713000820880940865610334943747 77`120.129684\
58381316 - 1.975643411546018565826831354237269647859413762195592200715785923 3\
9562626954102716240420464433325117393860964102350149256543432 92`112.024856403\
3159*^-8*I)*Sinc[2*(-Pi + x)]*Sinc[2*y]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (2.315024130210414398357549416437316520666903115608731640595124733090400\
9738083333135454025709729555520555240827541126581911117395836734 86`120.143252\
05819516 + 0.215337114900152269429916185344616630580581496409490010235200693 0\
4077983172533104690953134808437842744780037071509565506631685617151967 6`119.1\
1181542608408*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (2.333943929466167208047832749130888929830288521482310122954599756630127\
9627033448146855701873861867902868099467995946668812995136493734 71`120.144028\
62134754 + 0.207122634374707946204111469713649231150488898286153245152627 5129\
4859762921265696393800820508161045360050646570761824326753644677694 46`119.092\
1657641939*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (2.616958687575873496369178994304859718188572282925409218215463593540551\
1923725841885569518037345003618170331026336821862124833739522803 84`120.128181\
67701695 + 1.953741842607698707816849966279622783997863742788258624182126662 7\
5757509124167156690786282701087069042212032282917271278394714 37`112.001251988\
01652*^-8*I)*Sinc[2*(Pi + x)]*Sinc[2*y]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (2.514992463462014509189782465368413975321450767544325795494020787772026\
1762160055973756367293797047241862717672854812479752227890088141 07`120.129684\
58393696 - 2.164218965564143823712490473309314256384018175503900158497641860 8\
6547386859442088200358036621275336505811109742039117303224161 28`112.064449094\
5134*^-8*I)*Sinc[2*x]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (3.598733166956196148785791532804176846987108877773083433992619717713721\
4845356092625447675470817429337155428161184033386746695259233757 18`119.870197\
88375302 - 2.883282670235822418160900930903609215612114249505884487338712142 5\
2035634116237876888844290803487966607839881518158579726280208 62`112.773935458\
95349*^-7*I)*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (0.682641403312067512995718915906105567102468309835516238368960489132465\
8595743001546067829586294305253265305714755317369022833636704134 13`119.632869\
36944047 - 7.055375221129739008970908883578040618261357256112383413737779 6843\
6672670350715582432541901704491900956141535484924106084562160 7`112.647196859\
51884*^-8*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (0.736703088711129281805221075646099451166650747836537407062658489955166\
7387238995980543121370903759404310595010914308726487650918009243 69`119.664051\
308275 + 6.415114271426936138111513494251117020413992926314238336866919158465\
9523821090637722926151708554040353913370116747424367563636272`112.60396480606\
93*^-8*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (4.155891533038719504533364534161511107651472337799854191333331524802558\
8315453107192531899481292481419351367575898997195144352342859749 9`119.9086973\
060922 + 1.082894273754902978293175452534431821320313151612838735171055710302\
```

```
9062043915640291284824766937826631500629789391144542152924558 5‘112.3246191591\
6672*^-7*I)*Sinc[2*(Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.18634543003042137737945443268772615053625423825231377488545847071496 3\
5303498824062057754749914504070456910653879579022899155403350004255 119.682905\
96434494 + 0.215337140412885355604118780903597420679734488858177552425109614 8\
9857875000450425929704485966226293993555066254472788792385260223913065 5‘119.7\
4570615934837*I)*Sinc[2*x]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (0.68264133295941758940770579148225969701012386348385613314882424723205\
1333309256804069460238787827167339829846468074245409455797186149655 119.632869\
3379326 - 6.47446271179331981225071786967885940117644058459362101487904897840\
9428990067005226986594634467019895377912411575330195549709449155 112.6098804912\
733*^-8*I)*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.47065353365065267038010680117590889151370834579021517208228189639363\
0731427581469200496408197566101169243391098567019965070926262614195 119.836765\
54087383 - 2.9046614189682310292788800895903365269900729718086092870252173343\
8059308119771759213143279492469835741710742365908794700976955735 112.627159733\
43327*^-8*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.48347728133443570185074615513841661547719273082232192062819589570436\
8224949479268660550340074975816436887961509983916264844536750033695 119.881008\
49505713 - 0.0072644623277376078349572566454332914866994296077970667226541820\
1041388245872550511635023302133483136474342620158270092321361541195106 1\"118.0\
578358996485*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (0.83087501843168983883000352200230225993952320893330543465921491568757 7\
0575255934753852091814891129913665907811177123231819572175003315765 119.716368\
562424 - 0.026588696575716916227104192006304413815801425783339533745280833469\
7580132505242662388474826175401231913281996420116229411860438098507565 118.221\
529908988*I)*Sinc[2*(Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.21746020434810390844845703896673423197779487926710465997471157114481 0\
4104393099494939833603906329169922750737183308869185419179214803965 119.742274\
75782811 + 0.207122608876367299644106577036372717285153095517617227107816692 1\
2769906798859473724902876576773241557965035580722017723565087305647970 3\"119.7\
2112247377473*I)*Sinc[2*x]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (0.73670067258812735034028391518692838699534239543943508578428459920432 6\
6551345315662760473839982970289254520046449488978100638174099712 67‘119.664051\
49753737 + 5.879668808363305453246834783714572012086961316930466055365010002 7\
9204679767816801453724083508807268793545930593781310676750659791 112.566113294\
88646*^-8*I)*Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.43974637545026394912881990994376919434744709141559193105570787505304 9\
9363637349615980559765261672699394883841339891935467015868804909 72‘119.842093\
90012279 + 0.007264468160898270951168700311778623438885955407589199616388792 6\
3559871141229688922342398105248989879688578599439040997260207887744769 5‘118.06\
009545624899*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.46241164451331349818962251224371428045436961911599460981097361430650 6\
7746306839395508621460507423362572208107009179548229782188322880369 5‘119.820559\
71756414 + 2.358747913297228688511580774536883542613220687321179672060305655 0\
2430882544837210626416880300022545279414809673155006198250572 8‘112.528212484\
51872*^-8*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (0.60740790913953728559761653550979684270223767439602547827053859622407 0\
5936216610895135327834160645981779789719917938168061649851125324 51‘119.593650\
2926703 + 0.026588701696367132597659648217934408260350698335940567068360264 1\
4771550877288619990537083618366447711231216619210514406028320524831790 8‘118.23\
486698166481*I)*Sinc[2*(Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (2.61695900666562286675826090570245304042179635387020628786675877227140 6\
3080569248811742235211538240331336251677753537060244519239034603 44‘120.128181\
67804149 + 1.765166815119600328365532760239576979195138051687788243622764022 0\
4433074445575214734311731336330302164617572279466806394832113 75‘111.957170512\
52949*^-8*I)*Sinc[2*x]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (4.15589440274817525436526743320140154449595707544911297438935947786655 6\
1951290404895835745399193165637279183636977953547611223767970768 44‘119.908697\
4667658 + 1.082894519789546953085194210368753865487095515848109788233333738 54\
2974129789544145073244022773288612385863913584691911431922964 29‘112.324619118\
6253*^-7*I)*Sinc[2*(-Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (0.60740843249362943142266532444114845795157187792873368643889473217468 1\
2777891562679948072633994370375726640013038215256362955209453107 15‘119.593650\
68507867 + 0.026588600388552746017680809779076661477037117016172040073456 11\"8\
9662940472368515131919816514489590392733041234345584626571597345912970 44‘118.2\
3486534513252*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (0.83087464710147253015730449340279876652948433782800105035468637374498 6\
2126624266609394614483122731918055169934723777892983794958080030 7‘119.716368\
41544728 - 0.026588595722570835995287607180871690651435457195830423477889 68\
5656309217950192059484637217006108124864644286498672604918303280473 26‘118.2\
2152830878554*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (4.15589279361935098454947540347007938298694108201173866620767024006 3\"53\
76852842838475665004365994054389617297860883817263714390976659012 119.908697\
```

```
37601399 + 7.9144908976995918819923465378580585913855885388736101957725855555\
188712882380614366207770400390563300666260596857455248480475111`112.188456024\
10379*^-8*I)*Sinc[2*(Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (0.4447115898155768481592465924108745498015471711409545254122396957793233\
157979008130639224339342725901878828657708256638065736892243158033`120.1505095\
0257393 + 1.71705665286224149350254520937724799283906029602920777039629654761\
772530009086881204857810707562172235035267952685588319844931`110.737215679230\
64*^-10*I)*Sinc[2*x]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.1863455298018368943547649522430094111207892667308780530611157232697745\
633517254278569534492790445027078675909532845416895331701236639553`119.682906\
17827542 + 0.2153371128561362756426143648313586913554803336533933899903955092\
446261997053964590690647036310519432939359722470978456783988859920243033`119.7\
4570608517605*I)*Sinc[2*(-Pi + x)]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.1663542783361213704772500001247832323258243090951578380699856048571254\
288782015561751555406697768687459758459867533891069760067796630811`119.931437\
43623116 + 4.140125315929279066017588593800271059651671076983855829384048040 5\
268713382129290085959460529209833111418604928027690390635428 7`111.32741694842\
083*^-10*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.1684800208406599912308185622794600114575853607011743460869555790089361\
249806316000003078234796966901389456681362994881766536102077201148`119.958710\
03064028 - 1.1294265848447906383234031663507504593529844920740995427357339224\
364178160782902888250911482001682470354040097181074534083140 6`110.78501962917\
666*^-10*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.2174600758634277059152447148025191323924058779306470789463192712637544\
428609906489392765339771234664874644110458678825517878283130591169 7`119.742274\
48839693 + 0.2071226377057788283060924590008415601677510468894993031726830283\
254788228998446744746065762280949196108460795916235505778204786841546 1`119.7\
2112252139263*I)*Sinc[2*(Pi + x)]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (2.3150240025454269314066173254901532605674162553524845041025047054206297\
468814495687244750583719739630319743576080655662791095137534739881`120.143252\
0558098 + 0.2153371438930234230771338017199851981541496152544678005056457734 1\
306274145096518203456566762320092941677940604204440741329826107012725 2`119.11\
181550612164*I)*Sinc[2*x]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (0.6826406573397099755231646854727565852025938947851633994483930464666619\
032772257719558499344990522534057177294702648211299024915858017751`119.632868\
95198813 + 7.624092051835142996272267493585181720754264559139665428935047983\
341638961528690856806143717711191879316742101693178751815035381`112.680864932\
41604*^-8*I)*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.4706534404520160689770683968224106752774109571309608915905314475109965\
200884670566673944040186204345834705010594189628543585056135728932`119.836765\
55644487 + 8.901656731390250228922075127371158383131011285084062285612282482 6\
763397824964058699888049331571849791784716695192266950177372 28`112.1135351616\
0229*^-9*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.4397462885490059430879932776542738389427091880554888236971197267674284\
552678261658506683797508693571100221687119243285897002295241502637`119.842093\
68076272 + 0.0072644831054140372902296988143897022431364303558146793793089603\
330547496270680711264158283800744918734980585677683790808601793036438`118.06\
009621614555*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (0.6074081026359135610199041550006887580480379070038695715240460667060 61\
264314955165196239831829604495516309517060376639224869042050587111`119.593650\
41999507 + 0.0265886543632890946513179015541362199041493127420624727139800692\
718639738470994892766791374582134965890389430821542848046957255740247 26`118.2\
3486619751075*I)*Sinc[2*(Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.1663543171036941348226474869220344350137770798672811793980276042682191\
935345274478425008439310351495927836820829985762376276423301418899`119.931437\
38438395 + 4.740736639531861686131569284475626450081562158700381785120179882 8\
464827141632784536868833340708277056255344506008713651033425 3`111.38624913832\
611*^-10*I)*Sinc[2*x]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.4706533789068042299323418850823559519376501533894874109069132206740 33\
397253906100767634796254342786029486495757316491407928462485423271`119.836765\
30514458 + 1.1398428331173706066545209617268265892034502620754259464658320727\
291221631933861498174043823903781338844955673619480007348421691`112.220909097\
1316*^-8*I)*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.0520280533114745668105862751047811695092199653593001275374805767935 68\
966108062035030989145416523888786864423193442630743117526136466813`119.343605\
20350057 - 7.8159547566712162084020527946055687852237339693288058478547659349\
364497233783144481481301802315921870579729417636301882096459 76`112.5203496636\
8723*^-9*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.1056143211573643165679355239394425676947831661906727063095076297100 81\
251425229967088597143185565654719614990621640938802987126052663236`119.669116\
7951111 + 2.3702683523785631776303957187658649526835445377260181781733963415\
900633457682067685720681599847662555519918838520500569798571 44`112.02019150107\
486*^-9*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.4624117035618859165262137174816437852073458950650902869649965240256 26\
```

```
33690234623666888097334419353075315956734524379345004038710271747955`119.820559\
69057902 - 1.728326020708811274491056374803909607731621848126603796086437595496\
46378955607283250629093257002932716916946159144377486465452729l\`112.393156541\
59762*^-8*I)*Sinc[2*(Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.168479969850297911902446746019688383663196878071597619412457595528650\
82216563408023223964413008668051046747706297589693867265965911414\`119.958710\
05567344 - 5.288185204017393312261164215505263605134016019942579836940516323\
52371487254601107080539339751422142304825020487488754067070l\`110.455468436252\
99*^-11*I)*Sinc[2*x]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.483477241598778369057128881020877079641295343427682147311737012695753\
52008926157056765459843369531806426877780183639759134858709289916\`119.881008\
55730315 - 0.007264452865440592872124130154112902122734242897284086861842774\
12714241134115218158046470935709552967686363326241075469101692888144939\`118.0\
5783543189902*I)*Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.105614383703684988600632946061867699406686525645936908508385505494922\
37651976980169809234997719794174689055460186464791818617544695639\`119.669117\
02777567 + 2.370269040226920261676883291071708993011189070940716254311475849\
33772177260645194397142617753544884299705434851827698075593053\`112.0201916025\
7566*^-9*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.105614350955672048168028996749833892732345004779439359289540402272413\
17142789345836602792599299513811726940758283900215153092108180684l\`119.669116\
89412986 + 2.123564528552628879427022934273904731290619589857853156742899426 9\
56663916662369656485593617155457192765613586265041065147549509\`111.9724594220\
6102*^-9*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.439746397884507197962987885721209309001497552949002228588469553994488\
78193642359801272900399764282335226851439993071594004185161077614 6\`119.842093\
98619753 + 0.007264457444212934367725441065392661626933680255322150512541776\
29555923834983321347037751349483574183102077128924140598346415749722799\`118.0\
6009487948741*I)*Sinc[2*(Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (2.33394402882690299211817449655549267000149572679186651867613209915386 6\
69652629537841851095645412978738141895944952837389815840369622307 3\`120.144028\
6230966 + 0.207122606981418091155989661475102180210772671867559826142788667751\
21773972736862665193758340045154506976684363851349340008824106509254 8\`119.09\
216569001593*I)*Sinc[2*x]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (0.830874986949166829779255799413684748364337699802010976823374140504033\
35467024121022891209487373196923876201363808690191252942831765622 2\`119.716368\
57352799 - 0.026588637246521066280883658359803459405679684409674064238464776 1\
94481028936982482012412992777976910449011218701748532631343283374880807\`118.2\
215289674754*I)*Sinc[2*(-Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.462411745133724980535630651071788915183065298346502219216916590974096\
60438202377995903107381060484369511799252868639111369681577405086\`119.8205599\
0885331 - 1.180104042008490251727131796717747863513142508493755151548949215 33\
83664215509767239776996838752483389363565855587964032250601938\`112.2274513502\
1887*^-8*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.483477287354056611200469682514631772524528266821862073992255258273301\
88326524569394776416821427927956933665267635371769490701167009009\`119.8810084\
1181648 - 0.007264475541195025946317063844933070465924369494910029148825951 71\
96272333752144777627658387682979212769661465397147360359595426721807 15\`118.05\
783660094569*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (0.736701210123267150001633158941264518134051225971245712423114259923237\
05685213891436455896909025978873337014528876662466426683270454748 8\`119.664051\
78552608 - 8.981144925493989021820614739949701711411093788850371328769225296 9\
46857611316270531149574360781703126216420796968649586376501051\`112.7500921067\
87*^-8*I)*Sinc[2*(Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (0.444711589815576861461373619808411962831140677422627472091335623652035\
71187503429492985124818082220332357851857669468057553068397600781 3\`120.150509\
50257393 + 1.717056650576384135899896946213898310464887248253512236283504847 58\
80688241829376055364863887906926472226734618119603012050060 8\`110.737215678652\
48*^-10*I)*Sinc[2*x]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.217460075863427744408078633245011878212488850264062969755567879392437\
82962022255922295199736983930264314757705542463144929036381928199 4\`119.742274\
48839693 + 0.207122637705778806054373609263678969093329170658005174283773464 0\
31544204583480330085243746952366661591509093482883668634943667066375829\`119.7\
2112252139263*I)*Sinc[2*(-Pi + x)]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.168480020840660006421625742466618307930039485624237323969341646164294\
13574334814299106761617531186145287523424898615031124064145751232 4\`119.958710\
0306403 - 1.129426074641422805027031305546158900805999922441443325173417284603\
21114269425813968110635073248261168343468219880930560308502\`110.78501943298991\
*^-10*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.166354278336121315743738084010266471729793658438596036184005459827761\
56005256138511278890045926428664437301607285583650676501384436195\`119.931437\
43623116 + 4.140124887276783038494856574692884101170963191814070737630303669 16\
77483456492585562533671244382774835854871609677895840203867 54\`111.32741690345\
567*^-10*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
```

```
     (0.18634552980183691420979851754221659597920255187104986471918095302168 6\
99713529531712632917156404688399311287427620865383088347000869059 1‘119.682906\
17827543 + 0.215337112856136393292744745545721607513531365510425650442905441 6\
08389922193816703680750077732838014125241707521894500815088321953493644‘119.7\
4570608517607*I)*Sinc[2*(Pi + x)]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (2.33394402882690300078344913675069571150122050873971406794508903395750 1\
46982412739170126707491111393896125034020281162618825852514664683911‘120.144028\
6230966 + 0.207122606981418070448364876173214483380508676720851498227523805 24\
79260600921998560741063024584214994602730155217086577064242491052646321‘119.09\
216569001593*I)*Sinc[2*x]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (0.736701210123267506462639762809867792743104713463667200789086584550824\
85448180775388168551835938475900137570749964821144318063914421458 4‘119.664051\
78552607 - 8.981144905894312550656797903604675710828286584579574152765407521\
74071048271684081149550200127119075350876211406027419320118475 9‘112.750092105\
83922*^-8*I)*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.48347728735405651062291441127543086801224926516666625019157469690799 3\
87780633700012989143193713323803681397733645833780129004431625797 6‘119.881008\
41181648 - 0.007264475541194960573660012989412658311649604139432416684866495 7\
63614862953711675186083218948489722506007819314824113360554188132382272‘118.0\
5783660094569*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.46241174513372476039994062637170067010262891606716646125567539523694 6\
70436342256047535619525024373170744478051257687534018722462089670 8‘119.820559\
90885328 - 1.180104054740582955211656452409211379177450856020368016751071844 0\
92541886683622474816339080965704268713020589115665861958959764 12‘112.227451354\
90442*^-8*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (0.83087498694916680059890675480608912459043550110290096941959730643182 2\
16378115357819884001616855282081727507831145606827132333481194231 6‘119.716368\
57352799 - 0.026588637246521230586959256740795790577603081344736051215 8258696\
65263594015446763800755911337590358613989145788940076023432938207698602‘118.2\
2152896747544*I)*Sinc[2*(Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.16847996985029785699236640116067894738279168027657627632369917755019 0\
44280461657217382099524015457579145124851689067695116298375121969 2‘119.958710\
05567344 - 5.288184453954893110716455807769885507643440485766638418027161393 4\
90470224034763703609106968776352672108184167561775565982342 3‘110.455468374653\
78*^-11*I)*Sinc[2*x]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.43974639788450714287012588210594610668635717477125723828264687844323 5\
87515213272573534905378365511446983883090681122930224949367456973 1‘119.842093\
98619753 + 0.007264457444212963731975441381272391395537520635850345750718601 4\
81926073964179721701285131435333733795095023801676362616208388664540487‘118.0\
6009487948741*I)*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.10561435095567213681266702446403930037218449315497851384683185082915 0\
70034421615410185111834039425342339555916295527626172081429986666 9‘119.669116\
89412986 + 2.123564541351904389507842895464215614262378203975275893162846636 5\
51334643832664303946270033433370674697530089100995716873359509‘111.9724594246\
7859*^-9*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.10561438370368493802163891388142115408055177181106230604395529285629 1\
67777561973547124518657947422334688863456395378894232724664669476 2‘119.669117\
02777567 + 2.370268991599153336981483913976029251123552717598398703021711263 5\
21875695095848490875568229170438936165374996948007894738012375‘112.0201915936\
658*^-9*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.48347724159877828065367597025151941036362522721144968028117206854887 3\
24217825574300210164447449818463748722859335827353963551833170183 7‘119.881008\
55730318 - 0.007264452865440492869068674035487447694469340197199593530925611 3\
75685102154925413607917144773142546093123020936754601227199065529503865‘118.0\
5783543189905*I)*Sinc[2*(Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.16635431710369411630656845185865329503307373485595080408016890587489 8\
75389461253650100446640528430344068967663516382897716837085026687 4‘119.931437\
38438394 + 4.740736759209884643503367690648072943082188148137500603203091744 4\
18727783750788928704825968730516894567512624737519167208219 29‘111.38624914928\
967*^-10*I)*Sinc[2*x]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.46241170356188613786723237172168637218860636786289724617753407327661 4\
08780955360257205268456726027146145673180578762566724130593663795 7‘119.820559\
69057902 - 1.728326021648449589359670261748714437204152338902881478676272840 4\
04195888368538012150166009405615363809760041866857017941693526 6‘112.393156541\
83375*^-8*I)*Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.10561432115736439349627392013472678410259726627664494940149446182693 0\
21778009224217495651953005193214753491138547291175502461054088191 8‘119.669116\
7951111 + 2.370268424225766420643213399168545322466051938026132812772865073 57\
67292691909445719717451405907816539564036654837105612166018 87‘112.02019151423\
913*^-9*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.05202805331147451571220141249126799159624762674644376633350291809497 5\
29713390570260913214294527813152850838168225335358595878130994509‘119.343605\
20350057 - 7.815954754526632981711251907744603237438704489997695349116797365 2\
26965190087765486303814956166851906204048880574131729353956579‘112.5203496635\
6806*^-9*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
```

```
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.4706533789068044062440389268935268715107457480796557121743934488424126\
183316812061646995636154274186379012187311854095317839267035202507`119.836765\
30514455 + 1.1398428318440898387245599580038067838478343944050538672712429517\
594338924531187609772223383581388784941319247079412464953021759`112.220909096\
64644*^-8*I)*Sinc[2*(Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (2.3150240025454268274070089661297034971833150406090591332120666889960954\
637577750664787407989400040962573150449391614861494389313966833454`120.143252\
0558098 + 0.215337143893023434550103756138877275208761890184741499405249602\
4090254268675109923181103971717420935543911356049197665948228604745802`119.11\
181550612164*I)*Sinc[2*x]*Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (0.6074081026359130135383642270798278741766229265455183783698032249476\
5350696925377390871533955460794592475256626678049948689583719548003 6`119.593650\
41999507 + 0.026588654363289259251251228632887701879684651251364293040079504\
6704114856394820505439806930455592964181254308481594961700307582933946 87`118.2\
3486619751075*I)*Sinc[2*(-Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.4397462885490058525713753388465250270784052781358527512522051403886 66\
7408527067765545341007769374972958549742391783025957398268234658 22`119.842093\
6807627 + 0.007264483105414176607196041596704880334977808716432681719194818625\
1519740094258154539524123459722579455435036981139901347478425721180419`118.06\
009621614555*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.4706534404520158566991311058073287971893889387246083480855158544861 28\
2826064374250203916079865100620441723220180427780900535208476205 65`119.836765\
55644492 + 8.901656633038972896452330462731735576734670501601603840202121713 13\
0370111635184620894197615736919406481349885276314200638156146 7`112.1135351568\
0396*^-9*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (0.6826406573397092052536341858090443723363528704656406733140481260099 45\
2147054295237528258613635436258869195757123656236699926597166207 49`119.632868\
95198812 + 7.624092039141480162076147769347050061460315112121440816101881267`\
4899801523362446210407086292497010598111354356506523360430111 64`112.680864931\
69295*^-8*I)*Sinc[2*(Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (0.1298380504054700176564264868526172278886895479286412488853844846674 47\
0964501127663500425591886597474532700064010341803006526183855516 33`120.147660\
58893777 - 1.207279273237555970394152895089003026375514551108914411528729 3183\
2769520149570326990635170275111605503758618747864967144711 67`111.116066347916\
76*^-10*I)*Sinc[2*x]*Sinc[2*y]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (2.6169586875758739508896155138279477703174241961826117458571900154384 53\
1628919239711673602073616762508459752158486766598836793237752585 09`120.128181\
67701693 + 1.9537418376594625095002730399721969378268530658538782837190222126\
8899979131720975674615503134800109462491690268657601433967334 41`112.001251986\
91657*^-8*I)*Sinc[2*(-Pi + x)]*Sinc[2*y]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (2.3339439294661671138270037025044449006875115863607887633056926487099 32\
5749293886115387473170454954717551404986735417593455000103881876 61`120.144028\
62134754 + 0.207122634374707993008902347653632138427555386502856340641077856 4\
8587423423820969030258482081748956511366792612079263294508596446330286 3`119.0\
921657641939*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (2.3150241302104145813284157967271203185720644756883272244662940003446 20\
8422469956728369795771450484215985786970806937478384718532646023 2`120.1432520\
5819516 + 0.215337114900152280700307662958024980741572008992550269666133698 48\
3940689289606800036350336550020424489039193394831708253006786417069623`119.11\
181542608408*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (2.5149927706025906996421688190959707857564135924142966319803391971 44\
2895574388604930737686005141390211017479802299867479314655884233 93`120.129684\
58381316 - 1.975643400197296139706218218506787502059868343533992618997951417 0\
7830075791569682963596017363367781470872865745876170941194339 5`112.0248564008\
2115*^-8*I)*Sinc[2*(Pi + x)]*Sinc[2*y]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (2.6169590066656231254437460251498351924561358591391205763493638523019 54\
3003897241919864623465955067015557727868052748293293168321589435 79`120.128181\
67804149 + 1.765166820178738234535187556939333953442545656867752416329111218 57\
4636262480691311238626682068301408125404444443979995374581429 77`111.957170513\
77422*^-8*I)*Sinc[2*x]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (4.1558927936193510704169560852161399853280076854310762800562376218388 21\
6327748946066111077130185296990940284447603560159992447376390907 28`119.908697\
376014 + 7.914490920898847683028410075036153244221114287727330895021802248531\
0139904266952092665251856966049211077798536181334731000113078`112.18845602537\
682*^-8*I)*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (0.8308746471014728203551140292553068531963371121165920687547995939192247\
0055549059220914970542510801549043475317555168875497102668351635 35`119.716368\
41544723 - 0.026588595722570914517956408054771541705425030435584246273729171 5\
0400867061411081424596872374724601801761147610379709676433141440308289 3`118.2\
2152830878552*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (0.6074084324936294868462388737922696481401339833595866831362505781746 69\
5239188555124929336350988102610042652500587765501641324107430001 85`119.593650\
68507867 + 0.026588600388553058251616937764685577597298396568466952785617 6435\
7794715142902133374102339971909414568309533418825157032820656142125247 2`118.2\
```

```
3486534513252*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (4.155894402748174765842987807781832473398246926763583561172154861941024\
93259649807453200646450828750534449445001784806482625202191447067119.90869746\
67658 + 1.0828945191398611117849763344077007432696639601222890496415704694611\
07699936667854259341938967840533703466674561369295469529582377112.32461911836\
474*^-7*I)*Sinc[2*(Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.21746020434810398505374400180033997982538032930764952675269615084029\
80368987957774016279457748125977702148914413234521629816011246047511967.742274\
7578281 + 0.207122608876367251119375385135583369432982699955450805348174499681\
31535115790948255352312005071336697203011533896084449798323205030148671967.72\
112247377471*I)*Sinc[2*x]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (0.607407909139537104883196047580778171915116282689220855212082525491710\
53127269005979551590453266206565815154909189827677738879016510143211967.593650\
29267032 + 0.026588701696367014803194797334848907565026350994322170256234808 5\
55335198891006000185148769416450669415143879458183392561344819021353384'118.2\
3486698166484*I)*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.462411644513313721026531803430703120206347051681502599756741745485023\
01963417942194921735076431583319647791487706655473015684249067171196.820559\
71756412 + 2.358747918153339055999793033786658574286188299848999914892613699 4\
39156370995909211867130013225532208568085804050194136619820685 8'112.528212485\
41281*^-8*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.439746375450264115338343174607973135828342828524930220549031377889950\
05866403161282506529105462557755661081567441809489167294396557084'119.8420939\
0012279 + 0.007264468160898189090348957045416906246816571591814029894932312 34\
11662678931320499873282419048431155036669217455945663257527173718804 28'118.06\
009545624897*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (0.736700672588127249503517994027228620456188864527363544694014969608921\
24987740844569729695687759607053016944843621561516225219418448666'119.6640514\
9753737 + 5.879668795619053941159785879352990770011609904267763978228788933 10\
13408069944578178122447370915599246558662046772597895334583841'112.5661132939\
4514*^-8*I)*Sinc[2*(Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.186345430030421325910129167933972967060612168472808284745714805264164\
75240206546870017251842095885617289035851175620982307243249928537 8'119.682905\
96434494 + 0.215337140412885372222299576111797480839152969805167766210945663 6\
71042110544444829669354932363342940943128908096345711892244963006277986'119.7\
4570615934837*I)*Sinc[2*x]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (0.830875018431690201776175719044535720243946474101375771141614766943165\
44547492752738722777952478175679719086526785510317094932248350253 3'119.716368\
56242401 - 0.026588696575716841053415426079250205044619833249761282519010771 7\
15922401665941284477237379954245932121846633968874627283561080373702017'118.2\
2152990898802*I)*Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.483477281334435597975535557551235061164834221310650238109282385413512\
92903464810746981252568691900090647387226008074251873920480674097'119.8810084\
9505713 - 0.007264462327737547295769687549742204534728493486253477984838720\
12682096582088549752810971425162522590560433128349211661582780647963 77'118.05\
78358996485*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (0.470653533650652480209317167951072214205001811683547791483637583179952\
88526021477875329162499678731320054493685270252909229312071058487 1'119.836765\
54087384 - 2.904661425398899106422359509815314271046512870790382207345180300 8\
85613696947467745121561632172802424023099381518588115829763387 5'112.627159734\
39477*^-8*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (0.682641332959417889775132488056663006866853439867014678491280442450311\
33949479544339185285266498046554207363147107294016449127737172587 2'119.632869\
33793262 - 6.474462709036062739201772224595527388540275771889030663054082240 6\
01848489962354720375008509291572803301216957026324112606441667 3'112.609880491\
08836*^-8*I)*Sinc[2*(Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (2.514992463462014848399682636949068377097092195820359980931642624432640\
03402261978689538578020202597731883045247250076377597232644521199 8'120.129684\
58393695 - 2.164218966164524751429883756353171478683547582000610297677497 6414\
45968624865312875165552898190086817899851539994885622261009541897 2'112.064449094\
63387*^-8*I)*Sinc[2*x]*Sinc[2*(Pi + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (4.155891533038719183684984786529787279688023168560294239151205119681936\
52201076178806602323403317130115499949993784868448252888026156104 9'119.908697\
3060922 + 1.082894275817283037676000081105519406988623136664354696265850227 56\
02486479292040188519167744496830973494309692218383336078651999 1'112.3246191599\
9384*^-7*I)*Sinc[2*(-Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (0.736700388711129373991418837765411664615437822309042575819400879861514\
38959569117602105025542060341315887520641845124537216356758878521 9'119.664051\
30827502 + 6.415114247671655670076019004520170470943940310310479880522300 40\
77470213501322372686444031398272183488677919630614406378835170 51'112.603964804\
46111*^-8*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (0.682641403312067403927827436287079475065741169789432695384199940011705\
49520959255857205157337353412620294460708545744316955762489410601 1'119.632869\
36944047 - 7.055375207916364517270318664212468504300103338391930532563098 7752\
```

```
3571298609156188776083727564905114491622401562910909617148866 2‘112.6471968587\
055*^-8*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] +
     (3.598733166956195549799667495773774875231494517594984138262329090261359\
8465851575259355262410209726786217689266717858911536987774789323 33‘119.870197\
88375304 - 2.883282670228948823051987265104541539122443858587106783828353829 6\
5348780885811659998824717544948651737888964247947245206726334861‘112.77393545\
895247*^-7*I)*Sinc[2*(Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*Log[t/(1.*^-6 - t)]] -
     (0.3052179354581322647582745248471220385991254062749213588271428420151 73\
76655681700512049257417013606429177446497163422716642693979774239 2‘120.150514\
99783199 - 1.122858923099027581805596091835176151968322022700874749195867218 7\
9170003832908795759142155679200286091226498330926039940673096‘110.71623014202\
28*^-10*I)*Sinc[2*x]*Sinc[2*y]*Sinc[2*z]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (0.1298379009118337334423362054220126505544317062606088088707842235931 36\
9106296480726921417814016800572806704797950713766964331514774948 21‘120.147660\
61120879 - 1.495548245342711183606225030580242406862723767583269601476196689 0\
0343702046445872982553384327717495116440623207944943246512 42‘110.209059553199\
01*^-11*I)*Sinc[2*(-Pi + x)]*Sinc[2*y]*Sinc[2*z]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (0.4447111552586326584841559216410835925464801746866760414713385440937 69\
5363896621205525527615039903437984548619724911941380422278762267 77‘120.150509\
50255983 + 8.515417278855267037276019062134866035585894455751228603904013345 3\
9119786581589063175025542882900680523182923599693732668165 49‘110.432637413453\
93*^-11*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*z]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (0.4447111552586324613592360405974524253763042756675330368991219279 87985\
2648965361352997913418361288423036201791570521094802492513393424 08‘120.150509\
50255983 + 8.515418633321717586409722120384632406792241618333572955258969 87\
8751069731737252286988184679962124735701450520520453423958 8‘110.432637482533*^\
-11*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*z]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (0.1298379009118334172981401823961152599284051017675001574227924894 19829\
5954486290622410403987289646084057102971977611538378026504074977 56‘120.147660\
61120879 - 1.495547373504966651977207078295520737783738758904731493639230 3853\
5570718508128598066896312834637589750776856642642999668303 74‘110.209059300024\
72*^-11*I)*Sinc[2*(Pi + x)]*Sinc[2*y]*Sinc[2*z]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (0.1298378986028519772525183639556144210937525539008908822670851807 78222\
9156485588013809540301207868922161453438399548148777748929950400 93‘120.147660\
63639154 - 2.039875959031511822962014269199677669916383809614064925110283 7991\
8565316803728047859896701555900589756005016336727136931573816‘111.34386291808\
411*^-10*I)*Sinc[2*x]*Sinc[2*(-Pi + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (2.5149901707073054613588530532866619122773910484969534052459732928336 42\
4994969523009115252031671811260852511830374103818357012712851672 07‘120.129684\
58445206 - 6.136223947878966021906457069182196199619744390285449669665382021\
1082010361845060647036696382413321434277694065697391613811035 7‘111.5170494936\
617*^-9*I)*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.1863453280198516591139578537401383296877246879114089072099478733685\
1196134021550488591779076092576965846293389768499737545299314097 29‘119.682906\
14554579 + 0.215336918685283157102707846476320500602838157008189103581930 1584\
6881446589337872451798879616545684658267382303549278826485780860132748‘119.7\
4570613111086*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.2174598734805352054750664708965697422155487066833484860604317966 27478\
4927751339209001464851100424255492063168077226730346423647371292 37‘119.742274\
56301465 + 0.207122426190195737787901148241114491502464134452835487051105593\
9885322019150123521074169772195977288091300941090591782707617291011214 6‘119.7\
211225566882*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (2.6169561218310104863744806710673021772379638564563624395945431188620 40\
1991839753127932239111179198115613906782828291786842917085143397 93‘120.128181\
67383291 + 6.224437885410788084721634175106803957859481161055181372155974577 1\
9066672266181151691273821298672289593532653776416371106943871 6‘111.5044853703\
0049*^-9*I)*Sinc[2*(Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (0.4447111536285598822859629178183685143559592103397249035326592737 77468\
0363464214483353361595289823660038481838289603347787635180491419 57‘120.150509\
50256594 + 1.052115754330655272565296281679225816949382136750098919177021 4314\
8299057327908315356415093451148757376706905052563920016161992‘110.52449500436\
288*^-10*I)*Sinc[2*x]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (2.3150217520210787563968837964121620604165962489579958327291896540 32679\
4804716939053230665956840420900513125141534531800487137857737 10649‘120.143252\
05696667 + 0.215336918822913726019548812962070587657908414854068476921 7520729\
0168947628639676225555481290677058062295499392614717011628558258792 82‘119.1\
1181547554888*I)*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.1663541478717816153287028875429583769356201978265344705286888366374 61\
2676587728853966978095329348516536606995853085251244827979790836982 7‘119.931437\
49830635 + 1.113655756130029651632082263859555269074389388188764311615858 6033\
8904239879879846910420265821617997773058227813713292314464477‘111.7571548304\
1083*^-9*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.1684798347945057656038904362815491819189030544126701211965334170780 79\
9609627649809500153052820719036064037320530219194125571591318754 41‘119.958710\
```

```
04363076 - 9.2319682223801785635837011571495521062192656734281038190543410276\
37026333840157663656834546328392580169708497447474431177043l1`111.69745641656\
014*^-10*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (2.33394159842401683442333108164789888381573932720049263518954358025164\
9\
00792430917174995941650213841742708465575071010764476297985464168l`120.144028\
62111852 + 0.20712242663908708493557582403787713227014474750632137870817263993\
85130858858938530846145831127930379537250760221723694268955647386311l2`119.09\
2165762139634*I)*Sinc[2*(Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (0.4447111536285598435952501147179620143896445184966740385319800632326281\
4\
14276277889278443257521624165755475728037598761996807905007536251`120.150509\
50256594 + 1.0521157557286149264436479030026162036442592130669637452624824288\
98163577292366916026461090932573384737701538288411411605325141`110.52449500493\
994*^-10*I)*Sinc[2*x]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (2.33394159842401703058927915306207816843547599666914653414636994390263l1\
3\
08637160283598335414347029383307785658127799297238253275927442045`120.144028\
62111852 + 0.20712242663908705767494837162216789364648239181334152468354967551\
77989037490625582735507070402305005578596541683749408330717457345477031`119.0\
9216576213962*I)*Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.16847983479450571775239304350118057673128895545519478080383596935676\
6\
93139741728257405455645654346676529559697946486988977484135188016l5`119.958710\
04363076 - 9.2319678581345093001602385178746597722203452642781302353374186171\
78730495730921006947285473559696874894660721471482936427267681`111.69745639942\
512*^-10*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.16635414787178181757293649464854901875289475806082882350023211723142\
6\
06042893032172861840804451083261433706357409159368390663774995715l5`119.931437\
49830635 + 1.11365577012686286944203541468843646906640406550210035473275433\
2\
66037119079433848530849139201815933827262045495262925747336801l11.7571548358\
692*^-9*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (2.31502175202107857429674030694857604068438503607427405315401048145551\
7\
70938715004273980157922539330472890054359054414671068817915726913l3`120.143252\
05696666 + 0.21533691882291370341194648098436579460922405943645196745886450724\
6\
46876685066819450285357773731313675428603934806786668632834768928458l9`119.11\
181547554887*I)*Sinc[2*(Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (0.12983789860285198612489233937996893199922632911653009752983386301193\
2\
02177463159406668600648218160338924940063371456525853195408317474`120.147660\
63639154 - 2.03987594854987560336042653029799015497500362469808461135908325225\
6\
56263319185607343174031668876284717221671322283649242949607639l11.34386291585\
254*^-10*I)*Sinc[2*x]*Sinc[2*(Pi + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (2.61695612183101011462318132071019680632180962415641879423944611822506l1\
4\
45257498561766191863338592391010308065948239795933528904156461323`120.1281816\
738329 + 6.22443788092129242231532949487807986309609222076289667754554620639l1\
4\
69676301296853588892706440952030674731827096550387248240037`111.504485369987\
26*^-9*I)*Sinc[2*(-Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.21745987348053509874774143019776250152088871404251829408858855535646\
6\
50548952388014256369027955471885420113188569676781226806459159490l9`119.742274\
56301465 + 0.20712242619019571860946279799270243645617397972588660794072167l3\
0\
42008869072264293794294056174932121479805788078981084031264010464956l02`119.7\
211225566882*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.18634532801985175926444153067675326324101220233986301356662153131239\
6\
63550720113335816385875297298819566281993240236904667863924585073l`119.682906\
1455458 + 0.21533691868528316981297337061899862691422846080890399924548340446\
3\
90399915569931007915130401929710466810949531758722523153928406692702`119.74\
570613111088*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (2.51499017070730538260112715395368804886338377018436236786307837627665l\
8\
47899223624995942031164805338624769792066266598550210732361029l27`120.1296845\
8445205 - 6.13622390451281043693438593247001073457063230947695099768483189l33\
6\
75848885686929955660043047170942022031503588817784042973535l2`111.51704949059\
241*^-9*I)*Sinc[2*(Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (0.12983790143990136172812035716644127265181715009984091794004059378587\
6\
38934250701775460799382021540790935470710546948975277604120054559l4`120.147660\
63196039 - 5.21190758118310791081359291455050468352251992791137864482163206l22\
8425645865601286129232499145564143505727549047156829525046l2`110.751255850647\
18*^-11*I)*Sinc[2*x]*Sinc[2*y]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (2.51499020509704961120441306452309497001546784402461118030392278153523\
7\
04810170389560175816265571644272196900670817019248569764958279496l2`120.129684\
58369103 - 5.22351340867054440347058755089553276369926377241961363268741708l7\
3187449020297124860603329446430883732345098278819394892359976l3`111.4471109995\
7257*^-9*I)*Sinc[2*(-Pi + x)]*Sinc[2*y]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (2.31502177697485032135674442032661965594274186585200984021508831621961\
0\
80336611102182447234121834991499400675369102227440007959793916071`120.1432520\
5738093 + 0.21533691524896448378433528382577723067170366890844395235514656255\
2\
38895508741438200997807231364970327994475503555697951334448791930298l8`119.11\
181546407389*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (2.33394158418192658184661906387394238005256196456078608287211742123552\
40\
```

```
3332671902092989687079179369055125015544484760975119536714365151631‘120.144028\
62089503 + 0.207122430065109213620202895964016868875735454287286384915805683\
0233693368981305880131774937784960363794504485965236435688230853012848‘119.09\
216577174993*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (2.6169560624097014209917435045138252461780232391225918866349137901093446\
9850906480423515689184841391764974758495720637953456963067236452765‘120.128181\
67420026 + 4.780954328674690344883535166269454480159338728599235873791977573\
4632983042570192263242467810158350802891904041581248093763229311‘111.3898998381\
2382*^-9*I)*Sinc[2*(Pi + x)]*Sinc[2*y]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (2.5149901281689979630956446195910072285791472620985746150896320139004107\
6793872061183950029038489945610471506549889054242087148756029388271‘120.129684\
583721 - 5.345755538252449444719402649068334411931158547731653856046322819542\
5447036779423299172819699676260070392385660611661821747968551‘111.457157393576\
76*^-9*I)*Sinc[2*x]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (3.5987310263240837447899694544892177231759217909496685764651448606719101\
4479128657221546112651571252606534890697566629586739290135103322022‘119.870197\
98485746 - 7.341425820155356213020419014218644858587240884081684621897643761\
4333213776649041099391269182229104580834568600745570829095195331‘112.179829011\
62122*^-8*I)*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (0.6826406080269450952590315567757878968072902795114715612922253672496681\
2573752269713295427587962980520613033439674983183397089093366967121‘119.632869\
33276035 - 1.795117389853315122775416795467439414331045292410351744032565844 7\
8748049706437561550402547115669471958371321439713042829581684631‘112.052770067\
39362*^-8*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (0.7366998656946330929780316932716904660088453398771546655282333211384671\
4596704338438912277845886246986741768860574612134890386385123130451‘119.664051\
45082275 + 1.634555059046286169485154494675114109739760052206321365037193021 5\
6217960377197205993704626191982199078997716090704728693532815641‘112.010160414\
26509*^-8*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (4.1558879709118195874722638161602549862307476168320700319482634575274071\
3990580134964110309847244583868200724100303018847397799741557101‘119.908697\
33912373 + 2.756669134882366434909971185404087907624140051048477076554438112\
4145230018000199829850145153107848631693571030911372096230888411‘111.730418150\
91709*^-8*I)*Sinc[2*(Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.1863453157283025358648498761055334630264244206002275185159204239568571\
7557128615068144338199347942710370263563139289590617967545324644961‘119.682906\
11216213 + 0.215336921723248016635012617368414507927248712851467291062141695 8\
5162115425717066148516561337534637275976554061043171641971029587786910‘119.7\
4570613250076*I)*Sinc[2*x]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (0.6826405902244560085110774427224581432381883205305317214831822382491450\
9715114807116580431657077173708602282577433693119086661949944377441‘119.6328693\
2451264 - 1.651513893139569391308653703875929803168278501133312021681567187147\
0779052841099924698850693180007624172835197974106427580394240‘112.0165594478\
7535*^-8*I)*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.4706530265846688658873777871200512436862421106731415183775407594320788\
1425503708033994981062253891965186596470344459697935834633703342461‘119.836765\
59117276 - 7.386698694511038564932310090241822873545944792369160646476547991 5\
9\
3408282630239645969763479298913476068738909702862357532868366‘112.0325151197\
5247*^-9*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.4834767353636947364655069774674129220882406651363538803282108095782875\
1486920589692820387006187366146133572851989034247951356580420966841‘119.881008\
39812425 - 0.007264449368680261379079896545955082314965036762221576640313839\
6918375349738069291008537303532892414110780690567763304873635979859059‘118.0\
5783551816349*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (0.8308737519854409851577875053285305411081877435398548557546468361892120\
0114285664075586400405170406735806046725357361049630498966609612‘119.716368\
35807287 - 0.026588615385383372936678137967254347999642781890882924503388112 0\
6869285003019207716891984710833299893858327980431178305834597027857515‘118.2\
2152904045414*I)*Sinc[2*(Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.2174598899973234310656413905318223448496080665933295037397778153515597\
3651378677048342686441750589015724676990124410923210363912912503 79‘119.742274\
59090967 + 0.207122423551720614411511645100864002353252271922553503980119588 5\
5419273995360174086130958662822132185684851393556404834923717334727235 9‘119.7\
2112254606479*I)*Sinc[2*x]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (0.7366999380808906528980767871866532002501961183018265402472710830667781\
6519265062448252244736859771822804098347086091847998298138454126‘119.664051\
49878013 + 1.493898888718156148399101753716070532537429806577680999492851595 2\
1640696741379585353924227920823782178571721545562802794788582‘111.971082069\
62847*^-8*I)*Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.4397459640755827514515887136848374902227859321233047354117987046591 77\
7441752574479184070954149039195844592517983864389274596663000524 42‘119.842093\
85747118 + 0.007264450871271841157267709648463283360513151093067675023105322\
8811975477929816161812657107415478490366077193957693653490773402828‘118.060\
0947862385*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
```

```
     (0.46241124083467008830377392684958816137626542397676770830034952932408 9\
4313358719086715926464378835036564728544302689297601045286292876 52‘119.820559\
847598 + 6.0142273050144839602452451485615084991539318772247504691401819981 39\
0392158603118613368033015944782753114441178110880139038126 63‘111.934711302756\
85*^-9*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (0.607407761187540896905958076538135974891600952674076262215132841561 46\
6896025787695916838651114342114892983917482038585966839537559813 25‘119.593650\
59457836 + 0.0265886166273412079744229735470479291289933992952211966396213881 \
4383602333526471955703660405829940605562476430509727273820550536378855 4‘118.2\
3486599985551*I)*Sinc[2*(Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (2.616956144923212147301834623241958858720747241213108071304246292001955 \
6427805744455714634721367319289951736611874946995739998557067145 73‘120.128181\
67446073 + 4.6587127780788433217131741567360561557589468941857123218878388891 \
1850288535121162382940027612770042437346356963995449910049 7098‘111.3786511655\
6182*^-9*I)*Sinc[2*x]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (4.155888701555051703288464351314675315846630897291588184522434040154440\
2802598708823049850959721940823632942674851203827116755089361571 22‘119.908697\
38003211 + 2.7566692886884390841868064045136580123318041607621723345064189714\
6586421977744786198936568866745822919024050582057195787518398959 ‘111.730418139\
70363*^-8*I)*Sinc[2*(-Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (0.607407894326608232544941885749850516044727339172288052263571257269049\
7797606446149511162066556726898668920548972499572692348649912307 4‘119.593650\
69471396 + 0.0265885908769224211863508845948661081165394061714576947004437014\
1691963059041071579519563747814947067173646173096536299390650509063924 7‘118.2\
3486558419347*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (0.830873657333116916278940552780816274430471734523459623738842201681205\
9762110729217895805001152626927991711643428862756246408438080419 76‘119.716368\
32088879 - 0.0265885896644373475256149257393992505147462492519523453170527179\
3339493597394130769398994991693051163270410223170973241466769451799069 ‘118.22\
152863262215*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (4.155888291862233359878109321214313640092338881620675130998814223987609\
0480003493177580873498067057590954747071458632957124400512467983255‘119.908697\
35692621 + 2.0146817823721313553037243842114178055524133259329518896711351814\
7464808445193234106222684374320087429293770595858846218452196 46‘111.594239950\
86988*^-8*I)*Sinc[2*(Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (0.444711153940592865865212422156659114865090929376052539337661733164086\
1643578725529604135303091168718308530570565238752599528111385921 25‘120.150509\
50256582 + 4.3123950814409706896239275504267552710009871635628828069245354785\
6817011569842220557887089612349087311823408835122048086382 1‘110.1371500223314\
*^-11*I)*Sinc[2*x]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.186345341470663804579409431019767722109312244734085183833778324207850\
7587444623779211312049805722513972091003827814734544041467777939 75‘119.682906\
16731475 + 0.2153369145938250849327349547294525453535289845302422477076387904\
7444945482665976357868496658927978106373647354255421016137804790549754 3‘119.7\
457061132798*I)*Sinc[2*(-Pi + x)]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.166354136590872680176295460670431509840708503097302443067609210326678\
4958500012796740497415768239822253568939021590126509929638221588 48‘119.931437\
49362473 + 1.0956453266901441556908220131101418514490237650044361745787854721\
3804990501933940620203210342226645252586110846962309592636056‘110.75007387969\
76*^-10*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.168479846318544202228370592590003027049869171546179846189184425367078\
7102582774088738638577508424831833678176961145586190033585861564 16‘119.958710\
04632026 - 2.4590447306046326260480845868073955243413114122284385245728167006\
1690383156316244852264859960957605904271568884293078413866 26‘110.122928517356\
81*^-11*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.217459857624473477176147637634835823047384028059439366262098994573386\
9552745068820411624974619188334535902820469614061089242703840832 35‘119.742274\
52287305 + 0.2071224307785136017014848517216931720628246197945057436851323434\
8544839548881584755681312363428385392895868671959749122533444287403092 7‘119.7\
2112255783394*I)*Sinc[2*(Pi + x)]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (2.315021743571800577600548488921679810558628892461315637587659904187852\
1991718349127823276898506090588747925600888869156480773413096 1103‘120.1432520\
567695 + 0.2153369224714828582192922995004394471436560139465879221758955204 8\
3780652496970631937839714676038970320305461751830473611926166216528 4‘119.1118\
1548429528*I)*Sinc[2*x]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (0.682640420459842322297718817234718709054978993196255511048041986849575\
0395465010141305549325463983959291696284884777725719107171203011 02‘119.632869\
22678103 + 1.9429384960363651943644691073589767000199172409382119731353387823\
5421578607293181771566291161644873271484164208097410439603550 24‘112.087136279\
8313*^-8*I)*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.470653003022509924780422696638107213326774047259802249452779000152238\
3361629435494176638159233702213372308949928495575740094199566492 79‘119.836765\
59541637 + 2.1193144710852133013651622473373000874515498745623949229626297141\
1136477974633455800357617464753836166226496052685317051153808 7‘111.4902601649\
3197*^-9*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
```

```
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.43974594211677150646908850349311916425233274533867594767009788121282\
427231416458667241483681911814273939949751139524893132791819438091`119.8420938\
0192215 + 0.007264454520439049542046444796173477203132770320976445555920127 31\
612975854962477764797423441131464086733453410212860877210444994741734`118.06\
009497053606*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (0.607407812704482438974792628363674971765487906081087424067070948302360\
928249590205934331107312477143896799665067582456947604058448573611`119.5936506\
2768106 + 0.026588604624956419545484247611799479090676982044704588143362299643\
113066365725683956394826362158962848344044189378881952429996533085702`118.234\
86580007858*I)*Sinc[2*(Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.166354146945495693891301080628804382255828215149393415017357596202503\
445569000635285236463030153304204666055731385737100615290876934115`119.931437\
48188086 + 1.134580423857711864919819014462293794784552891311817374921105662 4\
816449705319645004379383821066072785698980304058245362435535 3`110.76523913620\
85*^-10*I)*Sinc[2*x]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.470652989496978733759114716537385746264532195132571793008783143435548\
335246552348553436144307289336859231447672629691628301595483077422`119.836765\
53360269 + 3.038320600174082543934449424333244656732405641311525704075820052 8\
192030046180825443699756859544442031966966730922525049682931 75`111.6466983102\
2535*^-9*I)*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.052028027236033344060111833043898935145864295842152231489103187064485\
161033157030423614065352335806852912120450335904380467888605612 82`119.3436053\
9017969 - 1.98943996011853656168972188248810077050901218291059699049025422858\
215849614607555876248311661958778535301223956098951238676846`111.926098868241\
33*^-9*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.105614216601848839432458959133999338122744107893092901946526074567849\
158073160568941509765106220004675905715643263408401901740034967 6`119.66911677\
985676 + 6.040259469602741679007654822931753470758234094138857455263634739036\
270241894364425416916541896386644373386367447242747105254 12`111.42644999274833\
*^-10*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.462411258179664298958898623812773869361150246959629672001478094050667\
351319719115948773193956628825710199775827707634622311705192067812`119.820559\
84320144 - 4.264213535396652674639806030477571946515194233369289257659624735 9\
717283801450788880058459253741958941085091260064299088828694411`111.7853703883\
8945*^-9*I)*Sinc[2*(Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.168479833820322958975816921409205186402289848099133638042745944197694\
545852724315495387517243891806163045154332417523137643594387899475`119.958710\
0541445 - 2.069695277791623943686966642559352221468174537501244318396514106533\
870291988818530044564005553314727491580571045920162828288 74`110.04806853745104\
*^-11*I)*Sinc[2*x]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.483476727830949704731369018071872884510136356702614686605154955962662\
904210999126324744661564375782442087987678343019080655164152126 31`119.881008\
41640532 - 0.007264446832048237812318712107679612483282672018412585168342031 6\
455185819731602584862898217417656781489800113511726723287171554144974 69`118.0\
5783539180764*I)*Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.105614232526475597686562355042957781818524776863621086430224816454962\
451570380912885258352607059761044767274025406629349158359987292227`119.669116\
83909365 + 6.040260038293767166857161760954663525176304064540330193866644426 3\
629328711202449373888151054853854866927333408070602261364144 9`111.42645002739\
069*^-10*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.105614224188652342417710243001566454008925718208017586618963453264981\
228570285913621907403812781587781180361398585904631706223028787181`119.669116\
80506203 + 5.412284013545433397472113419985037879681503585017985964992716833 \
759296635504858991720969323299509634862392966403376352114 1064`111.37877497032\
002*^-10*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.439745972098372274992870655570360420154824079520332696676321391332543\
973268199115583591848093931170241519681076910460395561200494745035`119.842093\
88201512 + 0.007264448270258544597773881942578116784723526028158683640965744 9\
225923880265286506142138268187811167764850817931710618861299313573333003`118.0\
6009464736132*I)*Sinc[2*(Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (2.333941608580828372291328208707162999403673319523498536079388888228505\
810395304920041434523930787299794011524933046486263372068025283 98`120.1440286\
2133586 + 0.207122422931411692762611170956866433263252719010272777984251667 49\
734021038363521890775053039821132209334705324544867917289921623650813 5`119.09\
216575269276*I)*Sinc[2*x]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (0.830873746221550878167816524486367212946549833625527532027552766715201\
516786614038011246738968051458398758452746382876730397463470193157`119.716368\
36118061 - 0.026588600230787278448563978412461178843670439979765572263468595 26\
807497841478607483015551869364576808013729923099593065339685621578785 25`118.2\
2152879904166*I)*Sinc[2*(-Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.462411266619871001643912051956618139262723450175438543506020485815006\
709207395882169619374186587493914190275814794172466072299894468963`119.820559\
89657687 - 3.151678252013033640067378431639314010587068151810059345812429673 5\
962324715006000132411383196264502810323608039392245574580125 77`111.6540733643\
```

```
7936*^-9*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.48347673733628190223836602872283383520878213905518539137861145416786\
7562342740763592290179912678355568688113303091773153783908814974777`119.881008\
37720604 - 0.00726445288869987085663471749942753362667617786872466625190611169\
0010837420426911793702847870148914245322148058844717166706857169429828 2`118.0\
5783570615799*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (0.73670007719191791309129391399371466395433702752709956266531539790061 2\
1400356025191895580481229197431968725012502435724462596462696992183`119.664051\
57253507 - 2.2848376374472428178855921512983581811924794076359860226701586634\
7465982800276124449267058341913897948337597297597221821986751111`112.155616201\
33268*^-8*I)*Sinc[2*(Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (0.44471115394059289525559925713902829977396208284409445020212220570096\
1360410607352865278469036288592100991380451562428481426493857447 33`120.150509\
50256582 + 4.3123950992615354780352352227031881107648235592307584735581965002\
7228036739812976276063008093666043839445298472497227724295552`110.137150024126\
09*^-11*I)*Sinc[2*x]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.21745985762447324937122715647907255599523420497534816727599302559839 0\
3548960752238486264546760005455968628037013418147527394737209652561`119.742274\
52287303 + 0.20712243077851359574888840184011618707461288743823522703341485 41\
5471007588844229459472955081172039612112305718772870796468670112717659`119.72\
112255783392*I)*Sinc[2*(-Pi + x)]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.16847984631854410913719220961158196520053360919251041759838124755100 5\
7202332864978059963053823324498063795809155231943105498909665944 9`119.9587100\
4632026 - 2.4590474634518194547808740368804261210631820896468116496824415154 9\
9571852623617978873033166883084109725943579731596001582894 4`110.12292900000756\
*^-11*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.16635413659087282851063788659228323244469817889344826421307251982827 8\
6827093939057235938982384813428149449042790144078027429884384672 82`119.931437\
49362474 + 1.0956449142196077771443475597691993672626844117937360104749119329 \
7576417159397871508236651797739473598703526202788478220099312`110.75007371620\
154*^-10*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.18634534147066401899848932126400460684132814276008553961250982401806 4\
5992389731874723632933510444442912362964314206598510232674004455344`119.682906\
16731474 + 0.21533691459382520488824491892537663150446942725594215727605460 82\
9815931084151858910625793200278288709693165937908385721988173036956079 2`119.7\
4570611327978*I)*Sinc[2*(Pi + x)]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (2.33394160858082841396519799138166515605405453454667935785120355218338 0\
6958915259574121806010950698148065955110742809620058415421904991 9`120.1440286\
2133585 + 0.20712242293141159524468872701564693463224825983551280754185705359\
9366921533928306004825761438499343201598540911771274733962451452798087`119.09\
216575269274*I)*Sinc[2*x]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (0.73670007719191838488732849900446993369818424490891723424771021116 9999\
0030825402561027731387940048661739751654728611724223643259579335 24`119.664051\
57253507 - 2.2848376292043797653267264396487586328354183666266499772131132612\
3659777593862648095517043723947374546513468957795589607983813 78`112.155616199\
76592*^-8*I)*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.48347673733628176247348113667509055574491673809644040688930887 8206912\
8749444049130242454212658350424798335782661502646973001335656595192`119.881008\
37720606 - 0.00726445288869981547473905783182582250603645579482492723242779 45\
2754458035638200293375467232637099922879835098397239371585784246519187 3`118.0\
57835706158*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.46241126661987111806891073508900324242900744257038418338366606728762863\
7441819656909480652959092115667246212551501522828211710686538138`119.82055989\
657688 - 3.1516784126130610704388263802558568448258028346507331310910987717659\
7994182052943281408143568890925474051948873795500230861601 67`111.654073386509\
68*^-9*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (0.83087374622155081262240275447390140904015530897982591862214707181017\
7889203051212365234514596719729611915285555698348748413769017021 38`119.716368\
36118061 - 0.02658860023078710026857979272969673554895484064954281897087106937\
4667014244388092135791248206674236183119724947369797519622251721755 4944`118.2\
2152879904166*I)*Sinc[2*(Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.16847983382032294018610431381298232020985750075600517829766040682762 7\
6028392149976790935135375469076174865880019119364586413997581899 3`119.958710\
0541445 - 2.0696956852647082242148566457853022283286603992060406074396127164 5\
6453156484380657473330523276204550719134208578842415303689 6`110.04806862295314\
*^-11*I)*Sinc[2*x]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.43974597209837194754146336142218243205745955937770421444885573156101 9\
9228237817269934017897208838704811982576941137375865913818300674 11`119.842093\
8820151 + 0.00726444827025857811313134764063457769534106339269010237773392 892\
9199893743613326561729887543419116601681916679372948084616608823023823`118.060\
09464736131*I)*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.10561422418865215938543665358943918607797775150249124588036421721783\
0718267337549092802314580338671472329996408550027089977648732001 61`119.669116\
80506203 + 5.4122839105502228982419856077704375571751046809856281915230692196\
```

```
417532192249596389675859314070653816051502152006744811919858 6‘111.37877496205\
545*^-10*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.10561423252647572614472923162381065493396441436241635094014294308283 9\
49856816253127505407190266855440669181930546021957080584883082197‘119.6691168\
3909366 + 6.040259986036811029864894641485293068657648467881202646527277966 30\
71788974598565586945870480541718934406589015649620666068849 8‘111.426450023633\
44*^-10*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.48347672783094971181592850780145669360211188761324873716063319226760 9\
64219291300706501757158932288652238548350045323670430441670271438 4‘119.881008\
41640532 - 0.007264446832048114021896184458139512527436888401472752472070777 17\
80251546339264735116535444404599153912100534114312388724683050498605229‘118.0\
5783539180763*I)*Sinc[2*(Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.16635414694549571803233475341042988902701994777114476094086661573321 7\
56045112683671736969883075059962617980007498066162260221500356640 2‘119.931437\
48188088 + 1.134580132814179445738181776685985481714448115601991523665 7801210\
63727977984624921380412557625006701256353340372510990847703 52‘110.76523902480\
292*^-10*I)*Sinc[2*x]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.46241125817966433197563490829347977313302402854265506892083324623 4079\
82134522541296634287293820963074685586605924285631068919412783020 1‘119.820559\
84320144 - 4.264213543290356454005826874886937858826285242390551463373 09888093\
24458995182106125498929026943452325235607983060620342433466048‘111.7853703891\
934*^-9*I)*Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.10561421660184880226407560426156189963392279127942183691148283896 1202\
83120660019752032530437438806545914835104991544022593333265316709‘119.669116\
77985676 + 6.040259781023470696762451465985786666715925039775227181674 09429883\
07994647307747843072608075305337582627666723158547102481145 57‘111.42645001513\
947*^-10*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.05202802723603351865560689677004896306290823726480207535390716 0669039\
74379162589961714561796222001264632849249747032868675240040077 8049‘119.343605\
39017969 - 1.98943991974194594752343264023808368232256929568716098739711052866\
04037450757627723120438143259846743732284958033752608938866553‘111.9260988594\
2711*^-9*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.47065298949697899014181519194592556107143553790068447427982156443360 8\
87648936294259266677781965702571075596014111831666094299480823921 1‘119.836765\
53360267 + 3.038320681814671420668701204589207793316444586924884153749 6815611\
58522671177800868431190374222009425622415731599148019144618383‘111.6466983218\
9495*^-9*I)*Sinc[2*(Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13])) + Log[t/(1.*^-6 - t)])] -
     (2.31502174357180056869536872727280195035376926959748618305728188577324 2\
14020031227880021569152877368738009654753353773122105500606844792 3‘120.143252\
05676948 + 0.215336922471482816273379895958362590621116674556174875105979978 5\
33545789334028927261189625913030404755344775074702341408489360156945355‘119.1\
1181548429526*I)*Sinc[2*x]*Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13])) + Log[t/(1.*^-6 - t)])] -
     (0.60740781270448234396402901465346030929040351436319612528642400676749 9\
76535799656755690669944269078080074515459717422482075433283824839 7‘119.593650\
62768104 + 0.026588604624956404593942017196512370425280077117592062570445393 2\
31648540823566920451521692323853182242720224403026456479803179061378519‘118.2\
3486580007857*I)*Sinc[2*(-Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.43974594211677149014033222399571978910602776484462539465497547728638 4\
02660530034216132618365262019640457510562930988981846996532394227‘119.842093\
80192215 + 0.007264454520439177782608906222979222432745006048073466922196273 3\
11064178801772388648185967346244305036441860705165839607454607084277566‘118.0\
6009497053608*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13])) + Log[t/(1.*^-6 - t)])] +
     (0.47065300302251014538751182693689938611149509576138880064394574219046 3\
74477863010651007528381633215270381132599349192004927580824150867 4‘119.836765\
5954164 + 2.119314390138397529861093389798408822432045049772822294553584581 03\
27111832403956637240070606987038213668153636892760161010889 36‘111.49026014834\
418*^-9*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (0.68264042045984174138147583086828030094552831345037760763181986402463 0\
09243974499241055585247981412658469214507355179896203746811225768 5‘119.632869\
22678103 + 1.942938507270512265189412457359893348048778717222847280831135648 49\
80101335728389723076198817951175576163336094115008986189053219 7‘112.087136282\
34241*^-8*I)*Sinc[2*(Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13])) + Log[t/(1.*^-6 - t)])] -
     (0.12983790143990141917323863169109288589927812608952439995868073870 4208\
46444934385731537162986221986080115772605580907129193470691359393 5‘120.147660\
63196038 - 5.211911086621925481178417875302958511247208022514901176988529 5820\
99044958136214997906206601503930491874837264876262632032393 54‘110.751256142746\
03*^-11*I)*Sinc[2*x]*Sinc[2*y]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13])) + Log[t/(1.*^-6 - t)])] +
     (2.61695606240970080948046157970930204118278552615664623860857815397023 1\
74337599960989500789374134975297985201315764934378664283483237258‘120.128181\
67420025 + 4.780954476353624504315944456344353007539610748337545965176824760 0\
97833285175092308575716161100761334698621150323269521595877685‘111.3898998515\
3876*^-9*I)*Sinc[2*(-Pi + x)]*Sinc[2*y]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13])) + Log[t/(1.*^-6 - t)])] -
     (2.33394158418192677779208596856129254297332446254992412042217527839851 4\
98024696849856249177816445752138930904127655140548611673947140313 3‘120.144028\
```

```
62089501 + 0.207122430065109296984077291334287386216118357243377965569471776l\
5431322201311245398075209996585829022082340831506693368248893807469608`119.09\
216577174992*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (2.31502177697485001991455771166720311841655925033790639502034255920461l\
882173854695674627547064531254722075588640143870353588665533880361`120.143252\
05738093 + 0.215336915248964509464976834574773603635630759282886515917676477911\
612607634303960807459031802549162744720963239936917620014491002987472`119.11\
181546407389*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (2.514990205097050261049111838605326873240318211712851603914365724954102o\
106819545788814324098773234835518871741472791072728304908350954464`120.129684\
58369104 - 5.22351364547754881128231034301244384097087068828690032965727655201\
798203108263762068782098841554048698993842482076191343490258361`111.4471110192\
6123*^-9*I)*Sinc[2*(Pi + x)]*Sinc[2*y]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (2.616956144923211990414549672501596615617334441169548988379432305651054\
141902165088010016335419890985696370308216063533674130742589103094`120.128181\
67446073 + 4.658712953189818700998041441610981174825745966639538108250904033\
932318800328831464494072504826284134683104402021281256956772931`111.3786511818\
86*^-9*I)*Sinc[2*x]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (4.155888291862233173111771076029301166690368491582926513453520760631718\
658675747883461843119568970005114203899509133544720910879959505695`119.9086973\
5692618 + 2.014681800740959447545590393673190510425553429306120125279821658741\
135292154563515583575741067594498937730262772592748247524802485`111.5942399548\
2955*^-8*I)*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (0.830873657333116366768307070672066115300532427060348910203591830528641\
643184436717902535614939200757339763282958245963032348564101434762`119.716368\
32088876 - 0.026588589664437202900345821442356333970969658365542586125440173\
649672497056911433552302290371457828983005726931058763712065344331025641`118.2\
2152863262214*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (0.607407894326608189045673673329780331141992175765700094900352503923154\
417331815812585580412910847026247291053009636262803882218189873207`119.593650\
69471396 + 0.026588590876922440652061219968274622360060391807256262945135898l\
650465384952323076203222595052220744136534463144240476381121240195285198`118.2\
3486558419347*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (4.155888701555051115545675942926583026118134832307743708380846270139350\
001189950888552777830182631270173097067788866436688632860195106823`119.908697\
38003211 + 2.756669292049454904985029574643015911663676782089393094168263929l\
986732855099089458455049039171459762772055972007735517799758099`111.730418140\
23314*^-8*I)*Sinc[2*(Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.217459889997323411368474341413168736228383160221230614619492994740975\
372272401032265872784012971945323384696986768518672819232081553395`119.742274\
59090969 + 0.207122423551720564224603096374972281578552219421881494007074295l\
832661536829758935392265069928784156337559600892823581975702271581606Z`119.72\
11225460648*I)*Sinc[2*x]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (0.607407761187541020884335677148993474329512031727300927705952366192071\
272438393785954749805697510198232838998156725676009667626801733851`119.5936505\
9457835 + 0.026588616627341367134933562802279231192617812745693057723511261051\
691310312693407128085006683576370587825668338600424674296638284544712`118.234\
86599985551*I)*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.462411240834670064278681572181508972160161633904654246581748391467328\
227361801728295735400663974995900292005595632167810573263725093905`119.820559\
847598 + 6.014227237561471192976321783837659258858994324260759533812712153756\
455556385992937087453468109923785888025879556983536165514152`111.934711297886\
*^-9*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.439745964075582596009778819836950049953730220557460050855807282237296\
886321203593304252588682576440817444615103325951666017712446061974`119.842093\
8574712 + 0.007264450871271847294539960735964400904789016342147188910625653541\
625676406413694502850736822839533152140795612508327734120683379091228`118.06\
009478623851*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (0.736699938088090655738941528437015226294122786968363809019697341620281\
020471256144188251307015979939118754400108572665073438644350671623`119.664051\
49878013 + 1.493898871556301717771942636176496355510135939708912003918645753441\
041697870125204635907310280916066470407906303221092698597461009`111.971082064\
6393*^-8*I)*Sinc[2*(Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.186345315728302577311297170001677547862166861340016356204573155061165\
401971679286886372912341268145236996077337273342523503457627220409`119.682906\
11216212 + 0.215336921723248022162055878480252580382755728119808350717675091l\
701462789483720626091316016473772749220324552784789851956662367396846531`119.7\
4570613250074*I)*Sinc[2*x]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (0.830873751985441101465119011130888976763503730254737119683319587959494\
091722320348403732753279010094788316768963886901947548781024424798`119.716368\
35807288 - 0.026588615385383259362546891999432477341141561600323930178759371l\
595492510098619678556751509553612273007619212190380498680449261873015291`118.2\
2152904054516*I)*Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.483476735636947455548169556892269581043150343703328562619035874987911\
```

```
947083664512597538729521691131912209479783170115728709944220751518‘119.881008\
39812425 - 0.007264449368680381201023779620753899631276020508363698340120692 62\
180293044536764795678488165943510881427210188743673609825639618590674 4‘118.05\
783551816349*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (0.4706530265846688238863973122322402328213935286563800400579032723001 64\
33644003569891380479539676846320336475688140187695939762569314234 3‘119.836765\
59117276 - 7.3866987375571024761105154348674733885049815884294159213639874371\
18638004809498283429197041098384003546129672864449956043776 76‘112.03251512228\
333*^-9*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (0.6826405902244559987770897256558708066827660969250760663742221725297588\
57253681272155486413375717583127722546163299539419365003498415094 6‘119.632869\
32451264 - 1.651513916950254532085204345455107881164437282087387641221048931 0\
30801727712057373281639029225822481246381438834026929066077973 9‘112.016559454\
13678*^-8*I)*Sinc[2*(Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (2.5149901281689977136987879188204319873662732087186061965841383813738 8\
61452677910081402407806223685919756958587896588534957743789245558 29‘120.129684\
58372102 - 5.345755349485051423990478311675611839768701322710155224848213111\
85121220264655169351397230834384743288890510350637755670203828‘111.4571573782\
4113*^-9*I)*Sinc[2*x]*Sinc[2*(Pi + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (4.1558879709118197602128850713947397486293851367016647813403144403218 08\
90626497320547000010834541889342496797311726945156854157442225723‘119.9086973\
391237 + 2.756669134241974060585838447298303179683279902890436110466916493501\
39707590424244309495266438626713185728542799625704530253717676‘111.73041815081\
619*^-8*I)*Sinc[2*(-Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (0.7366998656946325617402442245828236823121742671971142396692445007684 70\
06759850086693617486880593083139897704761754384307595053842511164 8‘119.664051\
45082277 + 1.634555079368182371531036133378763587006894682912883351969051949 9\
52413085490088505886947546045248448972108776862519623300328055 6‘112.010160419\
66456*^-8*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (0.6826406080269450663446643148051831543405807126343449930134101849053 82\
25748684619292102437475515634819667895544657636813278913251496584 6‘119.632869\
33276035 - 1.795117384008283664648738178422767590558297285588264711489110143 9\
04213226081173349748991848660934167439300559676317085970546754 8‘112.052770065\
97952*^-8*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] +
     (3.5987310263240833784924047770854189671582363969707989336939512587535 91\
64630003212091638852147736794875804756324900738731229583816677590 2‘119.870197\
98485744 - 7.341425826672244025290452208828039638174003366431889118087226061 8\
57753363573851956966735342355930807580242925175758282182053246 9‘112.179829012\
00672*^-8*I)*Sinc[2*(Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(-(Pi/Sqrt[13]) + Log[t/(1.*^-6 - t)])] -
     (0.3052185881001036777285703893620970266394757963926404014366298555172362\
80427169853566755592558467030764232206652668075046884410148976013 14‘120.150514\
99783199 - 6.302195401889410443899921110177678785816470613301347786245619549 01\
35260055241162114114676628657287869778544996445101302996561 18‘111.46539588330\
694*^-10*I)*Sinc[2*x]*Sinc[2*y]*Sinc[2*z]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (0.1298318681813881517274930603490438562428838443227240947503408688492 7\
10485222347337269006994619082386623601674190904674220779325724336 1‘120.147660\
44744226 + 2.186340188530183219209304898008029183762684971721036533632545262 3\
68140970417563348927933031922868493968639462202701463801427795‘112.3739757433\
7642*^-9*I)*Sinc[2*(-Pi + x)]*Sinc[2*y]*Sinc[2*z]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (0.4447120314223518592862324076626458294179743267616263243783195384155 7\
23128033895252915377801605525846614244258494400823738078373122469 1‘120.150509\
50254667 + 2.221862215631447601021041315038128282224054500828637567202210245 7\
14099980450390758231750927079020866297508490338168986316967 9‘110.849147746888\
02*^-10*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*z]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (0.4447120314223520074258291804584324349729221820068311357250774344249 28\
16368150253233208344546092156381456301514325146026684580723312362 19‘120.150509\
5025467 + 2.221862030902410439496222712864366517342681837507634692986191269 90\
72171749413022658251290234691018494314720633156416354540574 6‘110.84914771078013\
*^-10*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*z]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (0.1298318681813899236395059260671400561166637321534742769559585850266 1\
92745610692818022111997015803689822813185335701082500817932492445‘120.1476604\
4744225 + 2.186340231576519324212387054976043037460987939405461799559047622 1\
76508876728569688558077554806519295503313388295483409073946 01‘112.37397575192\
712*^-9*I)*Sinc[2*(Pi + x)]*Sinc[2*y]*Sinc[2*z]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (0.1298318181942537836463895021352868719506022971650630652794051714892388\
25960099512855322248448921529302112093767410526563593713308623912 8‘120.147660\
5888717 - 3.569778190188908458358426552275643528951977081187994849185651973 5\
64483655111985175333568712815086618791028137172044358894583 87‘112.58689939507\
678*^-9*I)*Sinc[2*x]*Sinc[2*(-Pi + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (2.5149951663347219332546197437130456699738553588351662373797062608034 7\
47302123109712084669246734181405751213087237135590831346247126072 5‘120.129684\
58929438 - 3.434527114449633053850104176073900186851072099434153264080457095 5\
92403671682580137540921012040310755868066886335309875229203205 6‘112.265014383\
92532*^-8*I)*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
```

```
     (0.18634565239737216543091436615950939856378511653718848093238288030219 7\
835573123641892023709572892509400014140472158356043817074802200266`119.682906\
09223858 + 0.215337335350819468314543829716638750663153957789871067289931605 5\
974309592988698831991408783620984545256501997728611048850070522937882291`119.7\
4570616214945*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.21746039436871970945321459696176030432748084163586340181313381185571 6\
653739795050003307877739187783935360425438473363772077445978311134`119.742274\
72090216 + 0.207122812579022120688760542433699369028321873892143861496214382 5\
692577547341824321510810800503264749521358902484660887166621278358368185`119.7\
2112248447796*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (2.616961688444719657770980174539245217843969904875016706116488723630697 3\
350265463198341937971874471775306504065590060721246762075427747945978311134`119.\
67789794 + 3.406604392578271101258506376760573111892843401118951164950423934 8\
172321918252869552292121218278202356909319481078912938860807723`112.242706015\
23105*^-8*I)*Sinc[2*(Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (0.444712017245185796768380961746256480145153339686625254058831131739403 9\
906164933988572539187672809153537415157543848051134204379566450612`120.150509\
50258032 + 8.470214870768436436388678310332135947017565750241182317988898085 9\
892851749583557733432212108175640585159321460319770452232044 37`111.43032506480\
448*^-10*I)*Sinc[2*x]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (2.315026313824762589607682423936923674482029878842183830026425903300924 9\
022102538266643500892223974316321158695673636283781895082728577 2`120.1432520\
5639484 + 0.215337336126088837254514929547782551900488342670495110206677109 11\
756218029330727459235471567072593023572764519234087010190601767773833 5`119.11\
181546081262*I)*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.166354486063591913842868808496763308599501834917172238593395921314384 9\
962734212099182414403463446856074958683931855314206366012113457631`119.931437\
41004288 + 6.159320086847324659223421689435719445661095007090380942553496984 6\
965326965831519222056870545727496843612626029965907981130513 89`112.4999356671\
5045*^-9*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.168480121097900879103542397874180884390463922194757670047255974989514 8\
831462124998844757475220828773707342896335132085530855476686842735`119.958709\
98959502 - 5.113684486442867019079650090127110386429913187805046969733574299 1\
067411744201059707309567357432694994027151043744394920942107 86`112.4408952522\
1034*^-9*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (2.333946336464519733429403628181501672126875690378414891251771958897411 4\
112973746391845627320222966417981944925050477883807769388825258 24`120.144028\
62296417 + 0.207122815076826131233130175576950323363175924059865167131594646 1\
235666966997006611994474066700488402854429219308766538006953498602015 48`119.0\
9216569681831*I)*Sinc[2*(Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (0.444712017245185825574561719268266599141338773445092172387851432531077 7\
745443106236102028697379954697871309420117365425708872056864215743`120.150509\
50258033 + 8.470214592953999238281988847600494109090244860993090548335570269 0\
398579469493366403161867433395878254656364609866928933764 4422`111.43032505056\
006*^-10*I)*Sinc[2*x]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (2.333946336464519366514204609295748558185471426687772082194767378630437 7\
096142832253571177941503232576523200049190946985708055767452748 66`120.144028\
62296417 + 0.207122815076826009141300906466271652980313473695532654423131182 4\
608451153868300121931157374941245264943743603984937159967554893798924 31`119.0\
9216569681831*I)*Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.168480121097901063367145005606995544842137398782090986798336027790890 7\
734394083553024633685057499845846916077795700760890475386802607947`119.958709\
98959504 - 5.113684461257235205746548332529555431629234103881129623307797 7949\
463887449711371718515713738737128868141119596643880088094056 54`112.4408952500\
714*^-9*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.166354486063591757212084495630379233332169228353076312416714304832883 5\
233076642945052800521902150204482941636422546439699445529548141067`119.931437\
41004288 + 6.159320085987367827909054521137343942535238653745511255734069966 4\
107930818799172234894668280944093163623368166212737607355693 06`112.4999356670\
8983*^-9*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (2.315026313824762865246208156546440182026029969105596834257295542889577 2\
242425246370983426772057815857194506249796771575607921792679768026`120.143252\
05639484 + 0.215337336126088883515893526816641252377502152127638453806293821 9\
994140528378113359699349921918105095411403181995506661550601323257727 76`119.1\
1181546081262*I)*Sinc[2*(Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (0.129838181942537820884089868511099312016290948417439741214367580680935 7\
792163157554629364999619305510436195146956445591599607070666903855`120.147660\
5888717 - 3.569778215291093710033528207800615670421296047940216409166648181 02\
434889968631875655533367079548775841818233286940077670785974 6`112.58689939813\
067*^-9*I)*Sinc[2*x]*Sinc[2*(Pi + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (2.616961688444719614777290547010335502448170141388677940268378807709880 5\
070970294980006908234610478032861850639591852908100926147550339`120.12818167\
789794 + 3.406604413879638525358400142852117065134505725733875769799198512101 2\
263329779546069163989784209676268522849259729821779050176745 2`112.24270601794\
667*^-8*I)*Sinc[2*(-Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*z]*
```

```
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.21746039436871987018712537775700545186229852395959776071775479429994\
8\
23993696727382912439682119271138030884979880377184092588835608679\
6‘119.742274\
72090215 + 0.20712281257902210167437442401149346746611256655824203645014264\
44\
94987193923196149685306886004500875101399306857427970871701286842348581‘119.7\
\
2112248447796*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.18634565239737210011757336121827762062564997083164012550476013735542\
7\
72587255078581768436999661604511741247985180736987987147373277245545‘119.682906\
\
09223858 + 0.21533733535081949892082440719517120723180466690033978519146076\
25\
20365556876176611582031112140630530596420449146790112606668373495954116‘119.7\
\
4570616214945*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (2.51499516633472198670370636632020280970582993875520835086891629335127\
9\
20144109376979977355509548702432849627154023976494949487632081599\
6‘120.129684\
58929437 - 3.43452710881104659204937731145749596702564752320703507547319339642\
\
67458878161770716472832696731452027761921964077120333254847550‘112.265014383\
\
2123*^-8*I)*Sinc[2*(Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*z]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (0.12983819773921250962287122106395173697650964571534480723124861992060\
63\
11120802030330958818710393463864985564962033947060827215959613322‘120.147660\
\
56351948 - 1.37934518391750530135350366784539411565928143707807153136627897\
47\
87396499546273307268900983552516370789477524037396256482188021‘111.17393104787\
\
168*^-10*I)*Sinc[2*x]*Sinc[2*y]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (2.51499532985328901867457699390622300773988080179094886404980875289954\
9\
92231272702224139850825155711913690175272657471020490636185894498‘120.129684\
\
58480155 - 2.73018652294460498492854976034605777120442820974400811592180978\
3\
16192410884688415006722272942162847715337548356558048794128621‘112.165339720\
\
47003*^-8*I)*Sinc[2*(-Pi + x)]*Sinc[2*y]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (2.31502647168200142371683784880337680965294106008739467237224082186950\
1\
19233716966387129798866305300880557170263015166932776685821619765‘120.143252\
\
05889897 + 0.21533731782712278193109584557966162087315954104691016254224716\
64\
67220679542442158829886296697361977192690907547389776635684862835575722‘119.1\
\
1181539679752*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (2.33394627738965232949007965315681341603268809805118182592547728349835\
6\
19515115768179217893891378045968725688215042938095740629897993151\
6‘120.144028\
62184276 + 0.20712283551951491550329655929482814542740051022855486722591897\
01\
72121346440998513170953710978075771254710409813628354186570404937286058‘119.0\
\
9216574955357*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (2.61696133219211856770820001642898762924023791822894759851585709592340\
3\
92252743452371602697586107202652117981368175418193596516039437405‘120.1281816\
\
7963985 + 2.80681354794356070007620495234828444493601839156728520394863519706\
13\
64940224974582277439360558939753801434552578057531705951954‘112.1585979381\
\
3623*^-8*I)*Sinc[2*(Pi + x)]*Sinc[2*y]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (2.51499489262175203502374956988432470219314504610283692485387454182016\
6\
95849781331461200330042744456260608706253847220335146453757160036‘120.129684\
\
58497944 - 3.08509920321075289019801552528850339349842086141737831145651376\
38\
22862916297189741062254810749540022265873826927068477140481774‘112.218416611\
\
13187*^-8*I)*Sinc[2*x]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (3.59873588837610184494581930580487027916970476258116290280350743076976\
8\
98074522506531341381308512640962772718559835465654923441095036563\
4‘119.870197\
82302441 - 4.06278730249582767514010563684679239608649114975727693095623249\
22\
77407500171126642288368461143904329729003117156645892107911581\
35‘112.92287193\
\
459545*^-7*I)*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (0.68264214850871690439474877842856573475339212879141494199416414886250\
7\
07311172844622454505801556131231327805099269185154832217247506313\
26‘119.632869\
\
3875421 - 9.93363132069693751807165681841328378826909529123177028999256114603\
\
19285645207161775030160647653729618780503706167329354106719073‘112.7957843254\
\
8502*^-8*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (0.73670101523188009321991167229982551885106935513232821149261425694935\
4\
49073770818319403865727779060009029146962683520826308026782266966\
6‘119.664051\
\
23187508 + 9.04799081475573964065802726765362461794236681058756263838006417\
88\
72966603124984307070723100569765044831217974574369638041136339‘112.753312114\
\
56297*^-8*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (4.15589536088263772582000904970772990201407482054414707121357559167383\
9\
63109001028763162634882452172352498602300514094583207364317132024‘119.9086972\
\
8786451 + 1.52607139966175101898453209241974427714815886261424034615265074910\
\
64892623064867836482726019448042958341349463721981952483079\
57‘112.4736075369\
\
6329*^-7*I)*Sinc[2*(Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.18634557813526726250115364990190779646498019626436268398998070034257\
3\
30919362582433828963917601919407766809939320816344835826607724901\
95‘119.682905\
\
89265527 + 0.21533735381919943691592244996470102844644581299389011696701518\
12\
82816659260482362626534269355763996115979029352210094713022141492345469‘119.7\
\
4570617288758*I)*Sinc[2*x]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (0.68264204903946264743498767173565077495722679308141946717191532832931\
9\
43778985436750078379958110519419996561166350490164121518320958063\
4‘119.632869\
\
34382992 - 9.11445406353325195551036286758428225784914509990090850082845139\
67\
74162840367030083246867600095928596153577240671734438597364940\
7‘112.758406968\
```

```
06335*^-8*I)*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.4706540208108550226425662359343919431434573682378548877604373130311777\
98769615792830821182526613852475324983220203024903523486251210666`119.836765\
50062575 - 4.0960832795399097553131381989688495514144803861209414471873647169\
94475988502368111532462778305880799079399790396677147341735815952`112.776432505\
31378*^-8*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.4834777987779870327902866591742330327902030231790195702130373711513555\
94617332075954151117411233138548036919399158832121162969097424864 3`119.881008\
54327097 - 0.0072644733032702529305525618736807485672335787003915616928740502\
17906038701183887532665560961276343904696076588196577848991471442215829`118.0\
5783613921162*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (0.8308761043752862348431423605871310658308856792635268153219075443843 41\
22873497315224695258320486615105712556781745340076482692585027732 9`119.716368\
68212029 - 0.0265887553902630166176126564766431553871650007685428862082764257\
84149035771950130338189335551385337590677613134379364086287568996455966 3`118.2\
2153042173109*I)*Sinc[2*(Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.2174604795482468841333629065326994888454513250102737535737808069070 45\
55052913089429413250535626404947517054346030985641524869650142139 1`119.742274\
86279555 + 0.2071227996303275483404566050269922388452908594521289413100327785\
69055946012359886051553758897790235980734493530119443033491615854644274`119.72\
11224291069*I)*Sinc[2*x]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (0.7367014149182188098637591396196097755264066091857043615989149587821 70\
17724103308026942547335407775204743941840810940268174694259839147 5`119.664051\
49926902 + 8.2941625311541683731789397998912668831208877188858798080193710100\
64497563761166395540220016903430875994521548848858510155408910 1`112.715532536\
78212*^-8*I)*Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.4397467960699206040764179576934253654810360925313877346696328218 67067\
85700155864431435542426830782546634142684711934811708550692014253 6`119.842093\
9158987 + 0.0072644814621985142015842019304404661669159818925381163542664870 5\
01437508538929313033420705165468297868362729105415274894870508011546 32`118.06\
009585181619*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.4624120717964181390212876459454540549686154757712675785633902188721 23\
24603906907699896837697090130555904866207521163221749107085728895 3`119.820559\
63487956 + 3.3207402949284013984674536255913101254356269251420813766206204427\
35996280900696776839302032399420045811184592291738536337010339 2`112.676765383\
63091*^-8*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (0.6074082579646416652813656532824175201531368073181015544673237543930 58\
08349096853609659919872855492210034917122816399883230390729467939 3`119.593650\
12260324 + 0.0265887627844680736037928708851737888054914155718096777244089189\
42984444338046073765146925506877536779823053285405215048915746203635721`118.2\
3486755998867*I)*Sinc[2*(Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (2.6169617779681923169496145133338272495246263222718109123337188037347 68\
36254631240185494296386539904757691820419955861311362555388118700 2`120.128181\
68108356 + 2.4519019361198124770960107595004371133707947924725909645102993425\
04958032806752545990826517688554931538352441078638054878934447 6`112.099887398\
05974*^-8*I)*Sinc[2*x]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (4.1558994046751217063803163242664420075943987167963367242057193566139244\
31127791019970909732425196814211327842474826105905743319811 7899451`119.908697\
51427421 + 1.5260718935629530976384047440172282186732342286440559979880754879\
92086320763514214380426671346820575101195803922694115945572219 96`112.47360748\
134956*^-7*I)*Sinc[2*(-Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (0.6074089957715252045624296646025929570709352954862194265480447900109114\
29773071765676737483326169866255058562521830230135114993816838184 2`119.593650\
67486807 + 0.0265886200224066470362654263066224884311622488418978891368790391\
34561938853439453735363286484185391234710342679417042649000816977820161`118.2\
3486525287727*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (0.8308755814561085992167785164057760104320868874527146557301565835777 22\
15581089656498893674643004202577315516694053677760793343289747215 3`119.716368\
47429109 - 0.0265886132816874657223442785613043621004316150003989253430164022\
61673232835080566659910320313276587548117039433647867517679251648233345`118.2\
2152816605474*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (4.1558971372042081969363939738733770516228145685124580778267222571110 42\
56714158806046046012999919093331371351730442682880951511440811733 5`119.908697\
38639329 + 1.1153726644490495422408329374258123657780122702652527270180584424\
48093345694005805167683800248488888235368145374516512334845873 27`112.33745259\
321495*^-7*I)*Sinc[2*(Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (0.4447120264101481296922322668171893301568243145791455217889169281209 26\
32979832535051190026854053325596090225311718604103470771477498678 2`120.150509\
50257991 + 2.4128467471064817062625891666767760769171117373821676496361069534\
97969752456530053315464319654994434720584679582848079180150 14`110.88496036668\
808*^-10*I)*Sinc[2*x]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.1863457176965725621123062976977798646813328919408735626018096137627 68\
98451296620387467040960930465453424119482734607706286313176625451 8`119.682906\
19201424 + 0.2153373153780797777577470752092408910756961370094948965300324319\
```

```
6806425253183102812042670437193476171233705967129788899842497903397572 2‘119.7\
4570606945856*I)*Sinc[2*(-Pi + x)]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.16635442675541291430045752311617615449562654818667199707205953852312 1\
2436964751543684313370796202080459669912200402347378478257662724 53‘119.931437\
38616047 + 5.625660880302794723929542953994854165351902221151185285714975376 77\
9175977741326610965523140861471217266206771368545528214655661‘111.46057657303\
001*^-10*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.16848018800524412259539901715377899915632101749645232534026334276404 6\
6659865323658550240326656075986320116978191028413980961814820174881‘119.958710\
00609154 - 1.800146808893179868678827011046294237049761599708234084990732276 6\
11910109893068114682156469655611087190609431901951545201110 79‘110.98746909255\
762*^-10*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.21746029746714029062211011957111914310617837679180472563837145071227 7\
9741656027534281699950397879632356123371611155647935301332550686 1‘119.742274\
48140866 + 0.207122840644611192789705234424344231230301348706939859413289849 4\
0384201449226928597176323920423239345971514543030055830183975582778302 9‘119.7\
2112249735642*I)*Sinc[2*(Pi + x)]*Sinc[2*y]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (2.31502629451269101568924354696214936484677458906915135121854309858469 5\
1415605168889167570608034654138416013240960560808298414002283345 97‘120.143252\
05555281 + 0.215337359149216184872202068070138205375139194433409560310701954 1\
1731053868417040786562348248480094357849702374188688888802326962673360566‘119.1\
1181551002677*I)*Sinc[2*x]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (0.68264109018449199657096460643850525849707184527618006437652476022860 9\
9129527045217829033634589338756371371216854339920020374803923857 3‘119.632868\
79837788 + 1.072630084372530473040549162372485185411537323376305321742024403 2\
6987831648068274419411943486496131340785488777358241778917304558‘112.82912634\
57636*^-7*I)*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.47065388897700573285172311078339423656466601938398313270721827527160 0\
3737391046020832011555960040775206420139005380539098652978661075 86‘119.836765\
52171896 + 1.296343743576597821623241651673826752073046976639294036864626531 1\
7145890030242732634386392307947102189427405433089180137254984 69‘112.276784046\
16657*^-8*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.43974667310987028420701821136795157511533995621039477550092943249108 0\
3694839290408608336415216020465262835826298114775470179666150356 51‘119.842093\
60587668 + 0.007264502971101039186360887161580019599714927038146371138422385 5\
0254861690080095562162593885442521292113746355819054279675773686994088‘118.0\
6009694910037*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (0.60740852380685275329676550765374320022677779361473189763132474045608 0\
1666631776984206962766469540784967971650136538854502702486101342 28‘119.593650\
29995018 + 0.026588695826643271845872557431415638589972718521335185544787947 7\
6457560670551276761735942525017707825996151913038892117631021990220558 5‘118.2\
3486645358508*I)*Sinc[2*(Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.16635447991307798380960586094592088695276341438615825772166412894192 6\
7558133178171291708459662646340655831941985162196171752594391615 58‘119.931437\
30867585 + 6.904669022832734167217561461973990184723140454323258010880024478 3\
3853938798023284164620456731255262427076165179768345186539023‘111.549545673188\
*^-10*I)*Sinc[2*x]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.47065379573823052752816974622379201998072408015872663857214660505301 7\
3449969545494091532416811557037824906644587827424277498468117876 18‘119.836765\
16101879 + 1.567893328150183497134759125029350651470227447636636180288748893 6\
1173331767943269448260335441138466432514524071578258710300945 82‘112.359380107\
5392*^-8*I)*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.05202809462254207820628140828206277830988985556616592342903868306502 6\
4698696702505728975354138735045113247760573613356208349464040065 5‘119.3436051\
3776954 - 1.101534564783684543742302436238712786194437117542070182368589832 7\
3394501987082108691660910684497400266446228105597692158110235 7‘112.6693653456\
7394*^-8*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.10561443098107596715343051027415407575404797362457634119713191918080859 6\
6511627766413066568727850442604976298353195544505250786419189488 59‘119.669116\
82907786 + 3.338366455430810939571310298873471357214223545679094085330527172 3\
9896597823346075602850577544691474537956639535364330607077634‘112.16892757339\
46*^-9*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.46241214798501875072718309935406661002558632121612251399301559140449 2\
8054284814025726863243587399071755041049366897507199063854115353 69‘119.820559\
58934033 - 2.473720956897886375987043105125981958878753581623523911249911556 7\
5723834554516870703812283668782713621631244237343025821831178 74‘112.548881062\
81255*^-8*I)*Sinc[2*(Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.16848011468235595364174895207917561060208575160227431004552716703549 1\
1228142674783776704875908828428479610070103504162042409955397060 88‘119.958710\
0369599 - 5.211462184346960873817146576353957794252927771396394140876841170 05\
0688039181336351860622047326828445490471175612675181332859 2‘110.44912097826561\
*^-11*I)*Sinc[2*x]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.48347773576682087533057378125271151318768250014153760821553141291167 1\
5479635203852592195678091087917630946144364519124180244298691558 79‘119.881008\
```

```
62359478 - 0.00726446032216071718850299335354631505595226555619629507127245671
9139942707217530387172718228867247439551263541975321757296311023031684 8‘118.0\
5783550008182*I)*Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.10561451911693740248968934603003484930113198488364718133980157268458 0\
24571610330360318015084534960322566682303720424701841015464743441‘119.669117\
15693426 + 3.338367743828134919860321768686121239713078110119891083190567361 0\
49034872157025654545441895869600740117207743055162578107290318‘122.1689277064\
399*^-9*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.10561447297075830817757804619154131024353226390087476390804687045492 5\
94128988474362573439334405093816068434131939813567335722115599416‘119.6691169\
6862387 + 2.990684071876988575199342294636224811064773235066347346110198733 01\
24821849112303698788663117418243173751895810661998276185131 25‘112.12116407013\
713*^-9*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.43974682066418142184198410707036498730633191765885838453879656765626 5\
97630986183437502101122511190305181005015100485130713950370178511 5‘119.842094\
029205 + 0.00726446600841732008805893025888769668452945318144078198772831894 \
73392769400246381842854537079904835772581635164179836689090947583501 3‘118.060\
09501695455*I)*Sinc[2*(Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (2.33394642012899776397158988281379682869495002068844530297528600434365 4\
27263460680508624534513660312207911303477932006688589293475423511‘120.1440286\
2432298 + 0.20712279738620408705019185044785371629924991252005321823019350824\
55188048904857615318579763960750845177179983817265428739114041532844 71‘119.09\
216564551541*I)*Sinc[2*x]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (0.83087605319305761503401717251202664291927835765458848605692672443710 0\
52259761762757188853221267514757649441464541978317615790624518405‘119.7163686\
9692168 - 0.02658867204773352532040413980310619637739233699060903964967437 48\
36273321645804066729996026402876509976117129741629951046712291343157 9‘118.22\
152910198571*I)*Sinc[2*(-Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.46241221307870260881156490975146853052341324571195238016718288127 5665\
49533142490644946281520506136625619211729648429117125193714381359 1‘119.820559\
90359284 - 1.62094149549855032198755484743489143737292403041967183170829726 62\
93807993635378859167916701164086676823365769417550277209953135 7‘112.365297947\
76457*^-8*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.48347780675551694949450443132205936415698725228079537209261699864415 85\
17161376624619418585987058486065398456431806841314589655549854963 7‘119.881008\
42513697 - 0.00726449147254816690871958336396619893881145656214731054690105 06\
65414250351936208971730367976416770863294808533487222004526027445438916‘118.0\
5783710013041*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (0.73670216555557370941151375485929414421799859110809127213116484222389 5\
81426655674800182499712648434420684565183746067968896031108512193 6‘119.664051\
90377995 - 1.26726445811711467358753730872061460017719845448393607819607870 18\
20988942849731529095142117136936411895949237342705287875858267 56‘112.89962721\
205715*^-7*I)*Sinc[2*(Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*(-Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (0.44471202641014802922206287998429902662708506598616917955993333288344 6\
83451779025020261101942591800829249854310046428691127103517445243 9‘120.150509\
5025799 + 2.41284665422359039151444599230327179389774879669699947214232952890\
69548325109059668489795640363399460486238720598499280599958 5‘110.884960349969\
82*^-10*I)*Sinc[2*x]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.21746029746714059515038482031701496398488385835572343398743580886243 5\
05220652566163231718398030519993917291538341610567927007710253195‘119.7422744\
8140867 + 0.20712284064461135207724627118916979050476730033773752304395137473 \
33489732847225142376223853726210571058579061823796510987933100785442 04‘119.72\
112249735643*I)*Sinc[2*(-Pi + x)]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.16848018800524432744901586254398104929158486062688089381185350356589 8\
42862549943642618495805638098723299606074384749030678962787978839‘119.958710\
00609154 - 1.80014680743132066336030120380805625331848228930366222790700 5199\
34194050984748945594439366045802099923636210900183452338595 8‘110.987469109891\
78*^-10*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.16635442675541279584198693492312082201050731886512560558949518560281 \
25351122140875982328445805305869252728481427045102095234206192035 6‘119.931437\
38616044 + 5.62566091127493504401935156787685193459876251526679142707137500\
11374803466692924597016618098082972701599688846966770916530 51‘111.46057657542\
102*^-10*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.18634571769657240707581848964712710427719996688064288582849217149942 3\
18593388725302043038878146645529259175963887539995821686459862681 7‘119.682906\
19201424 + 0.21533731537807963647139755796313497286351018205482184728 2100944338\
19448405844841024657301315127000241952373492441056664879601650663804 85‘119.74\
5706069458 56*I)*Sinc[2*(Pi + x)]*Sinc[2*y]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (2.33394642012899757736579902429888899010871198005290057418212137964771 8\
41785537529988842905073436755220995336243040657243377096282852609 3‘120.144028\
62432298 + 0.20712279738620409486922902777038416115178964248225823217572147 67\
16677641091730671439817343024349731866264799513843723915131520720289 148‘119.0\
9216564551542*I)*Sinc[2*x]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (0.73670216555557349183171789150699778420478890799029350317152543077498 \
```

```
01493298408497181247923170035325897403208931983778264310213009618 6‘119.664051\
90377995 - 1.26726445903556712215748458427057674011803487268753090505946067 37\
87640719706733054869696416558243486182828909723837403873764056971 12.89962721\
23719*^-7*I)*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.48347780675551735840385807104206718488332725356131678118252355331202 54\
61058010043777206839072417821684128127272221374263868756370037331 1‘119.881008\
42513697 - 0.00726449147254823199487388910250732648992206147827336591987965 81\
89023511301241257977681071405924902811873914215097877082458028792135538‘118.0\
5783710013041*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.46241221307870244622094610280020927469231288109363745791762705447718 9\
06847902495247925264867676230605692439648445614118037738414505102 7‘119.820559\
90359284 - 1.62094149270152542341676074037957360631957660024274027086475433 06\
16160786210495705029425141601690958422480529861294576527809153‘112.365297947\
01516*^-8*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (0.83087605319305888445475324887748908308593486345530211425113457797476 9\
30793289506520563844506121713099331941436359722638020291647383308 1‘119.716368\
6969217 - 0.02658867204773356956523201383551545012003616132978981654762472549\
55187064619428099015078164396089484725859256852692543879329040000163421‘118.22\
152910198575*I)*Sinc[2*(Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.16848011468235596267889725423403180420307667654424978843589015085436 0\
96943496390777193962249573931279632844816928512591130299573696441 1‘119.958710\
03695987 - 5.21145932709846665964555569520041809411849071412256124649161112 04\
32518124606836558594292853052895035423746617274130231960619 2‘110.449120740158\
21*^-11*I)*Sinc[2*x]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.43974682066418154925868512271412182104960619987908958616405664409927 00\
41233545708744769315911928688862314242396517905585238645930404623‘119.842094\
02920496 + 0.00726446600841745937721713798108615517578690146468964148970247 57\
65925432555290306180471341358136891500310677976808062998469861967432242‘118.0\
600950169545*I)*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.10561447297075846431545598881833032893138850205288311226786609748364 5\
79637836697250744254752205139704986590699184608996331127611755486‘119.669116\
96862385 + 2.99068404650504306976929230422212702995928031178036755596385840 49\
52788846695970232242650476932163785989240185142511686666625785‘112.1211640664\
527*^-9*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.10561451911693734498968983619208166375655734654661033792853778870 3328\
94432271411204950205223159451663297066453967558846055077775829563‘119.669117\
15693426 + 3.33836775228697116345414872826055195409415274825931990601427 5769\
71577458743402779608868080366069052406395082299015013172120198‘112.1689277075\
4033*^-9*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.48347773576682084276712188980495546731157060775513562827997632283 7946\
83562071106074448521382107476374419433652779269919292909168913876‘119.881008\
62359478 - 0.00726446032216070706303083112001522647198598867801438137714045 57\
07697733286753683737752288670876872109111700353823017696405811571398 75‘119.05\
783550008182*I)*Sinc[2*(Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.16635447991307792697012696217070531787476904425836597418975835035 07\
84854292636453694716459354444266535911756697830126461457586087264 7‘119.931437\
30867587 + 6.90466898874907460534631192500711454980296655463849552918178153 62\
02219906047023273083039718165408499449821991431713455825098 08‘111.54954567104\
42*^-10*I)*Sinc[2*x]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.46241214798501890475080831544028839633609378667100664606202419745668 7\
17721420353758659862661335803868319488974712919484970232497768263‘119.820559\
58934035 - 2.47372095047000917746276155998690478587355278302368885404622756 91\
34211919980508701931704736562949817396470543301891620780285057 7‘112.548881061\
68407*^-8*I)*Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.10561443098107611651924574928030820401274353235154523733616314867843 3\
32237379292619463547484880891501591257555379950374867015799901167 3‘119.669116\
82907786 + 3.33836647411550178927580464843479591223931964410365619834771020 94\
94663488816243948936765438291818870399006093461863757225299605‘112.1689275758\
2534*^-9*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.05202809462254200418922758156073609114158722042071824694256187441837 7\
89859988473269079434362695016017096486127789499670606247826055325‘119.343605\
13776954 - 1.10153456290748622046838964943984966194667961908051586742723481 15\
14713916135559087397118486198400150944821681589704868016668296‘112.669365344\
93422*^-8*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.47065379573823057737370880522070302960839537577522118067673227528824 7\
64775945381048932384053675820710109754336640657087284095151718599 7‘119.836765\
1610188 + 1.56789331367287438412253862760135717686730174317088814400875042547\
21681922776624526606730909898757540511722121172668958166350296‘112.3593801035\
291*^-8*I)*Sinc[2*(Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (2.31502629451269085327394008068606273962023893691673798488405477718503 4\
24929542161092714587876052784989103133403994562400476048141442682 7‘120.143252\
05555282 + 0.21533735914921625420596156532445645527454244743447558888495544 91\
22448414075423229176218976777630771220519530884130486587331318195034159‘119.1\
1181551002679*I)*Sinc[2*x]*Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
```

```
     (0.607408523806852565908750741591246807749666556583988379269080777805130\
5602762337323679668765346867048663972141950450878068952024918938851́19.593650\
29995017 + 0.026588695826643224117996532790353904955083789700961237476754429́2\
3994512141677946375858780308200458547640156901064971154137822045018988́4'118.2\
3486645358507*I)*Sinc[2*(-Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.439746673109870555017638167639318473058686981932633604933836420269389\
3456958803199184665416245760653359684823420573969587304470669689́11'919.842093\
60587668 + 0.007264502971101021862738650438385757232894157924382524058́3287863\
3716384489630512754729966272463771307196890898166868873273925219103́0343'118.0\
6009694910037*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.470653888977005463658666876081017859814024694304707779873365314778061\
1327290057404824707822494615277447760161548866308297323868526́85326'119.836765\
52171893 + 1.296343744233626797424791197478952691814384244355935463184́1345556\
3714447299453246897428437984217556690017399510753716318859033́09'112.276784046\
38665*^-8*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (0.682641090184492908809079411392284604269492064114699790900985́028913421\
0044364291048241302751932096606615285319530152164759493047860291́69'119.632868\
79837787 + 1.072630085593226609613577993041953198680504718460507499́2369971717\
5573677328146885026946527368949324764098475334935999333076116́536'112.82912634\
625784*^-7*I)*Sinc[2*(Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*(Pi/2 + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (0.129838197739212431975584825054734447521802098396511241169282́494128624\
4933639412984203329135991775550947082937982887475221867421120́0328'120.147660\
5635195 - 1.37934529655596422539305549315822946448302865491442637097831́6226337\
1769121484019943737521984103823043443584504038455896426171́57'111.173931083337\
69*^-10*I)*Sinc[2*x]*Sinc[2*y]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (2.616961332192118258743946195435185539657472864328238760́0054371500177́75\
6662018894593919726913427964257654875559361388699570054682́4225136'120.128181\
67963982 + 2.806813544459995634810333084978237502046158742182456489́0326317150\
7952117608958108596612758709594040744608853380176154218980867́85'112.158597937\
59721*^-8*I)*Sinc[2*(-Pi + x)]*Sinc[2*y]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (2.333946277389652376178239892071774003069178056819224102177433́0345327́45\
6786114411506524991457284854835968654091371281598318238521493́59305'120.144028\
62184276 + 0.207122835519514976486933287053255472271914386324388869239́6154871\
0074176343100493564402366857773283279789619271743046474479121́4144291062'119.0\
9216574955357*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*y]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (2.315026471682001417060622643107175896518309290561868079882́5934501878́49\
2332216016238410058479344414731153547255790438566925618́31209416664'120.143252\
05889897 + 0.215337317827122742083636891282569091561655180350602́8243916871́819\
7584700949927224147305297847498014767125225118746535533777313285́1806082'119.1\
1181539679752*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*y]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (2.514995329285328952343645352410219677588044303666476391429021́0762828́13\
5757782814631682584093858795575106784425269248037692682896810431́819'120.129684\
58480158 - 2.730186518979951295732245260085828871432378529873914023431́2420931\
0455857367859309160210498783692623903139695428572464341234277́15'112.165339719\
83938*^-8*I)*Sinc[2*(Pi + x)]*Sinc[2*y]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (2.616961777968192787954293508059229325249082026629026231062́5243933175́07\
8933219144267044114935610502174237732094034297134565857850925́03596'120.128181\
68108356 + 2.451901970325512101412117612680822640214670677981538575607́3738898\
5874374883358497178373636499700848150405233916093729929́21843474'112.099887404\
11843*^-8*I)*Sinc[2*x]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (4.155897137204208546738291899089105809065044855894923106273́2696352́18052\
3525301566676340506372197945049898667005852992946149536471980́42936'119.908697\
38639329 + 1.115372666679191726175926554519932009936574359970097́8388618291309\
5749478756446937182280610495942741797605669647377119006934́49785'112.33745259\
408332*^-7*I)*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (0.830875581456109613861620830907496602853299410527699041́5147752692́77292\
7533844488549089489406343421587287222407173863495914387439́54162466'119.716368\
47429109 - 0.026588613281687226147008973357443739270289646106428577́5381670186\
6329097056370768699136834936471070116993921796262931371271́521908726́5708'118.2\
2152816605474*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (0.607408995771524565882199337689387178699103941990090́3534238666666́12956́3\
2444204705187498021360305957073628193007904389593267188920́20178826'119.593650\
67486807 + 0.026588620022406691704530552492550655573797588080892366660́2701259\
2185583083216494603326232853189549004561905718981048971218147978636́0374'118.2\
3486525287727*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (4.155899404675120742490169491853687100626447543645117669826336́3968465́25\
0014865937677293292897650961134307860510881445990125109146301́74125'119.908697\
51427421 + 1.526071892913206291627909693754923323885783900188794́1512686014450\
7464652715934097355188974775018449723483640722834620449501465562'112.47360748\
116466*^-7*I)*Sinc[2*(Pi + x)]*Sinc[2*(-Pi + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.217460479548246950371037881963984874509797823773843479961́2007161436́51\
2410257655501850456236712692872596665100000242363988633145́60763598'119.742274\
86279555 + 0.207122799630327487089538586491993415057035683836970548́2091582880\
7682600114231261153181850168182418266839429106191988889286845́8742136757'119.7\
211224291069*I)*Sinc[2*x]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
```

```
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (0.607408257964641130356186194563310702354770027094312857209318366430463\
962215875998369228429173746824550866091654552006358651647405295042`119.593650\
12260324 + 0.026588762784467855942205170633687438852571706267688658904192176\
00210624778362947865834940519559962565584587124207407393027899113196 61`118.23\
486755998866*I)*Sinc[2*(-Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.462412071796418279853012238287715082878229056602529656650302178157 44\
931735288982763542417242387329803163139610430786048641147014435406`119.820559\
63487956 + 3.320740297974657090831110785846559460360077105649841272798329634 0\
656803641469300929101420479873643544962718908103577869802589346`112.676765384\
0293*^-8*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.439746796069920314799953774043290820434814664880821598693069641784359\
009925511195175181830691200163208140058606032412376341438651879656`119.842093\
91589869 + 0.007264481462198488021038099795663360627391006601504772101406292 4\
029949082031178084098618130350273197285208852393124829248139757329565 86`118.0\
6009585181617*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (0.736701414198218747030071022895342742763881773920464529222428912591263\
961154096432573815729479398652317135244236526140743855036463788276`119.664051\
49926902 + 8.294162548007378902154345659160834852545232535617455221413039859 5\
478862077980633653716864334835461613125840411437478039018638 8`112.71553253766\
459*^-8*I)*Sinc[2*(Pi + x)]*Sinc[2*(-Pi/2 + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.186345578135267350699948879661735784488500326218150570104192197048176\
802500105941219524881954917200677317572398451453359292668391630079`119.682905\
89265526 + 0.215337353819199455430727336744203085546272752474974204829140772 2\
943101970395973457919449536520660100442439078015247720625047691484043313`119.7\
4570617288755*I)*Sinc[2*x]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (0.830876104375285942696230774024727363124888209997841175201053297736551 0\
966941484170036477609409840860262113017667803160992427151222475273`119.716368\
6821203 - 0.026588755390263181264332209442895407031755336417035555614714653 40\
599334604015104556329742667832571694698196016123221091447870067903659 5`118.22\
15304217311*I)*Sinc[2*(-Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.483477798777987055309209496050295667381800171221379764980682522578332\
110766231006555045260420819048755250839873507901127224447195366877`119.881008\
54327098 - 0.007264473303270224207383213578374161363902617473448883957028923 6\
943493311104166882062502054416318229683674784255676798240623100755067 77`118.0\
5783613921163*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (0.470654020810854570093949745192887827545462245793294812641611905460422\
193836856661335812881849312248789961687960756461363918084868320699`119.836765\
50062575 - 4.096083279203560278508249744171277381305672488535846152378574096 4\
660806955983447066769691713076824032041744273041993025513829264`112.776432505\
27812*^-8*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (0.682642049039463078759112629630840646048062034006293265044689235210866\
897962477709549793509945919317980122846167435738078668342646344427`119.632869\
34382994 - 9.114454048481103328955594379938847451783740698681580038160255171 6\
339226099703841997507961700722554158743319646209384799847760 02`112.7584069673\
4613*^-8*I)*Sinc[2*(Pi + x)]*Sinc[2*(Pi/2 + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (2.514994892621752625433858480020937329769087578852122288662657691799200\
112016800623789102950306888437701081777691232221361177909126027025`120.129684\
58497943 - 3.085099176288256025719532764202801602297950283828090834086265461 7\
497425210117329462282737415787231206298588503271773297263003225`112.218416607\
34194*^-8*I)*Sinc[2*x]*Sinc[2*(Pi + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (4.155895360882638296548496653586504040583595365424016750251290430273 73\
035945623283364162648447323226525701040022870486610141767393575 1`119.9086972\
8786451 + 1.526071400250817832326707855178922824508886620919255331313962795 36\
690239520810958834569083865822060198588100338957855674036687173`112.473607537\
1309*^-7*I)*Sinc[2*(-Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (0.736701015231880878138553018675221482526530257321473957689214640943329\
745534904023113750321229068717139509064976769958231119907455171944`119.664051\
23187508 + 9.047990845479498868184704775317739710198429637387661035479943611 6\
877592106061816540828473734125705068402768548384475466424896551`112.753312116\
03768*^-8*I)*Sinc[2*(-Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] -
     (0.682642148508716182719709821647466634888114454282262844561737646990810\
312400721171928815449983020846535286256195076230527078924926949165`119.632869\
3875421 - 9.933631317465629550115876670970959002739343343442767771654094948782\
121456884383618897575266963316694105746883008038322423793245 48`112.7957843253\
4373*^-8*I)*Sinc[2*(Pi/2 + x)]*Sinc[2*(Pi + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])] +
     (3.598735888376100111675377827861292157140828623230793786772081621874767\
709444162107727068343367350954559175340832395024567193976670964908`119.870197\
82302441 - 4.062787305450733427417385230512537435443874586702672862026481307 9\
779552958474159960356126536488124488578271463086070064568785729 6`112.92287193\
491131*^-7*I)*Sinc[2*(Pi + x)]*Sinc[2*(Pi + y)]*Sinc[2*(Pi + z)]*
      Sinc[Sqrt[13]*(Pi/Sqrt[13] + Log[t/(1.*^-6 - t)])]],
  {{{{{0, 0.005744532383467384, 0, -0.005744532383467383, 0},
      {-0.005744546691155198, 2.7209164190222429*^-8, -0.033406037070265064,
       0.006794304679397165, -0.005744546691155198},
      {0, 0.033406028501733576, 0, -0.033406028501733576, 0},
```

```
 {0.005744546691155196, -0.006794304679397165, 0.033406037070265064,
  -2.720916418911993*^-8, 0.005744546691155196},
 {0, 0.005744532383467384, 0, -0.005744532383467383, 0}},
{{9.735818174219 27*^-6, -0.003399138381585317 + 1.9451118843991242*^-7*
    I, 5.980396567743163*^-6, 0.0033951641706343465 +
   1.9451118843991 41*^-7*I, 9.735818174219 27*^-6},
 {0.0033928398726404286 - 1.9451118843978536*^-7*I,
  1.4301733269896 69*^-6, 0.019751956480763 59 - 1.9451118843734337*^-7*I,
  -0.004016519321051555, 0.0033928398726404286 - 1.9451118843978536*^-7*
    I}, {5.980396567742634*^-6, -0.019756417205623 33 +
   1.9451118843891 42*^-7*I, 2.4239635499104 66*^-6,
  0.019754280957926286 + 1.9451118843891546*^-7*I,
  5.980396567742634*^-6}, {-0.003401466933934184 -
   1.9451118843631592*^-7*I, 0.004019386383226745,
  -0.019758745578803416 - 1.9451118845816005*^-7*I,
  1.4423988939827633*^-6, -0.003401466933934184 - 1.9451118843631592*^-7*
    I}, {9.735818174219 27*^-6, -0.003399138381585317 +
   1.9451118843991 242*^-7*I, 5.980396567743163*^-6,
  0.0033951641706343465 + 1.945111884399141*^-7*I,
  9.735818174219 27*^-6}}, {{0, 0.033406033205311195, 0,
  -0.033406033205311195, 0}, {-0.033406032366686655,
  1.1496333627754508*^-8, -0.194265028716419 26, 0.039510700111155834,
  -0.033406032366686655}, {0, 0.194265024955036 07, 0,
  -0.194265024955036 07, 0}, {0.033406032366686655, -0.039510700111155834,
  0.194265028716419 24, -1.1496333628858867*^-8, 0.033406032366686655},
 {0, 0.033406033205311195, 0, -0.033406033205311195, 0}},
{{-9.735818174219 27*^-6, -0.0033951641706343474 - 1.9451118843989548*^-7*
    I, -5.980396567743163*^-6, 0.003399138381585316 -
   1.9451118843989 378*^-7*I, -9.735818174219 27*^-6},
 {0.003401466933934184 + 1.9451118844002253*^-7*I,
  -1.4423988939827633*^-6, 0.019758745578803423 + 1.9451118843032146*^-7*
    I, -0.004019386383226745, 0.003401466933934184 +
   1.9451118844002 253*^-7*I}, {-5.980396567742633*^-6,
  -0.019754280957926286 - 1.9451118844262843*^-7*I,
  -2.4239635499104 66*^-6, 0.019756417205623 33 - 1.9451118844262716*^-7*
    I, -5.980396567742633*^-6}, {-0.0033928398726404286 +
   1.9451118843655 309*^-7*I, 0.004016519321051555,
  -0.019751956480763583 + 1.9451118845113815*^-7*I,
  -1.4301733269896 69*^-6, -0.003392839872640486 +
   1.9451118843655 309*^-7*I}, {-9.735818174219 27*^-6,
  -0.0033951641706343474 - 1.9451118843989548*^-7*I,
  -5.980396567743163*^-6, 0.003399138381585316 - 1.9451118843989378*^-7*
    I, -9.735818174219 27*^-6}}, {{0, 0.005744532383467384, 0,
  -0.005744532383467383, 0}, {-0.005744546691155198,
  2.720916419022429*^-8, -0.033406037070265064, 0.006794304679397165,
  -0.005744546691155198}, {0, 0.033406028501733576, 0,
  -0.033406028501733576, 0}, {0.005744546691155196,
  -0.006794304679397165, 0.033406037070265064, -2.720916418911993*^-8,
  0.005744546691155196}, {0, 0.005744532383467384, 0,
  -0.005744532383467383, 0}}},
{{{0, 0.005744527287594969, 0, -0.005744527287594969, 0},
  {-0.005744537441177679, 1.9309229351673953*^-8, -0.033406007130284 49,
  0.006794290669661 87, -0.005744537441177679}, {0, 0.033406001049546 32,
  0, -0.033406001049546 32, 0}, {0.005744537441177679,
  -0.006794290669661873, 0.033406007130284 49, -1.9309229352771 831*^-8,
  0.005744537441177679}, {0, 0.005744527287594969, 0,
  -0.005744527287594969, 0}}, {{6.909111451498 6625*^-6,
  -0.003398554748120 2812 + 1.3803662110828634*^-7*I,
  4.244042763168939*^-6, 0.003395734411969 7245 + 1.3803662110828634*^-7*
    I, 6.909111451498 6625*^-6}, {0.003394084952961 5577 -
   1.3803662110620 65*^-7*I, 1.0149358426852221*^-6,
  0.019752908736525097 - 1.3803662111522862*^-7*I,
  -0.004016923980558435, 0.003394084952961 5577 - 1.3803662110620 65*^-7*
    I}, {4.2440427631618415*^-6, -0.019756074331108424 +
   1.3803662110369 542*^-7*I, 1.720188105222856*^-6,
  0.019754558322681 88 + 1.3803662110369542*^-7*I,
  4.2440427631618415*^-6}, {-0.003400207226272135 -
   1.3803662110620 65*^-7*I, 0.004018958617977918,
  -0.019757726682111 67 - 1.3803662111522862*^-7*I,
  1.0236118311995716*^-6, -0.003400207226272135 - 1.3803662110620 65*^-7*
    I}, {6.909111451498 6625*^-6, -0.003398554748120 2812 +
   1.3803662110828 634*^-7*I, 4.244042763168939*^-6,
  0.003395734411969 7245 + 1.3803662110828634*^-7*I,
  6.909111451498 6625*^-6}}, {{0, 0.033406004387484 25, 0,
  -0.033406004387484 25, 0}, {-0.033406003792346574,
  8.158479319116 54*^-9, -0.194264936988 13607, 0.039510634036213 58,
  -0.033406003792346574}, {0, 0.194264934315520 47, 0,
  -0.194264934315520 47, 0}, {0.033406003792346574, -0.039510634036213 58,
  0.194264936988 13607, -8.158479318012 18*^-9, 0.033406003792346574},
 {0, 0.033406004387484 25, 0, -0.033406004387484 25, 0}},
{{-6.909111451498 6625*^-6, -0.003395734411969 7245 -
   1.3803662110826 77*^-7*I, -4.244042763168937*^-6,
  0.003398554748120 2812 - 1.3803662110826 77*^-7*I,
  -6.909111451498 6625*^-6}, {0.003400207226272135 +
   1.3803662110679 81*^-7*I, -1.0236118311995716*^-6,
  0.019757726682111 67 + 1.3803662110999904*^-7*I, -0.004018958617977918,
  0.003400207226272135 + 1.3803662110679 81*^-7*I},
 {-4.244042763161842*^-6, -0.019754558322681 88 - 1.3803662110765446*^-7*
    I, -1.7201881052228 56*^-6, 0.019756074331108424 -
```

```
      1.3803662110765446*^-7*I, -4.2440427631618 42*^-6},
     {-0.0033940849529615577 + 1.3803662110671981*^-7*I,
      0.004016923980558435, -0.019752908736525097 + 1.3803662110999904*^-7*
        I, -1.0149358426852221*^-6, -0.0033940849529615577 +
       1.3803662110671981*^-7*I}, {-6.9091114514986625*^-6,
      -0.0033957344119697245 - 1.3803662110826 77*^-7*I,
      -4.2440427631689 37*^-6, 0.0033985547481202812 - 1.3803662110826 77*^-7*
        I, -6.9091114514986625*^-6}}, {{0, 0.005744527287594969, 0,
      -0.005744527287594969, 0}, {-0.005744537441177679,
      1.9309229351673953*^-8, -0.03340600713028449, 0.00679429066966187,
      -0.005744537441177679}, {0, 0.03340600104954632, 0,
      -0.03340600104954632, 0}, {0.005744537441177679,
      -0.006794290669661873, 0.03340600713028449, -1.930922935277831*^-8,
      0.005744537441177679}, {0, 0.005744527287594969, 0,
      -0.005744527287594969, 0}}},
    {{{0, 0.005744523898888 98745, 0, -0.005744523898888 98745, 0},
     {-0.005744526484050362, 4.916240124160756*^-9, -0.03340597616795204,
      0.006794276659946141, -0.005744526484050362},
     {0, 0.033405974619761046, 0, -0.033405974619761046, 0},
     {0.005744526484050362, -0.006794276659946141, 0.03340597616795204,
      -4.916240124160756*^-9, 0.005744526484050362},
     {0, 0.005744523898888 98745, 0, -0.005744523898888 98745, 0}},
    {{1.7590992572972645*^-6, -0.003397497174748252 + 3.5144911802326 61*^-8*
        I, 1.0805575723769594*^-6, 0.003396779100852105 +
       3.5144911802326 61*^-8*I, 1.7590992572972645*^-6},
     {0.0033963591391757628 - 3.5144911804245776*^-8*I,
      2.5840852176450 23*^-7, 0.01975467081815901 - 3.5144911800144866*^-8*I,
      -0.004017670731205961, 0.0033963591391757628 - 3.5144911804245776*^-8*
        I}, {1.0805575723724215*^-6, -0.019755476796745795 +
       3.5144911798023*^-8*I, 4.3796978333326133*^-7,
      0.019755090812208102 + 3.51449117982023*^-8*I,
      1.0805575723724215*^-6}, {-0.003397917905116128 -
       3.5144911804245 99*^-8*I, 0.004018188761633151,
      -0.019755897494740927 - 3.5144911800144866*^-8*I,
      2.6061747855439034*^-7, -0.003397917905116128 - 3.5144911804245 99*^-8*
        I}, {1.7590992572972645*^-6, -0.003397497174748252 +
       3.5144911802326 61*^-8*I, 1.0805575723769594*^-6,
      0.003396779100852105 + 3.5144911802326 61*^-8*I,
      1.7590992572972645*^-6}}, {{0, 0.033405975469619195, 0,
      -0.033405975469619195, 0}, {-0.033405975318093875,
      2.0771956573394124*^-9, -0.19426484481054948, 0.03951056796092696,
      -0.033405975318093875}, {0, 0.19426484441300861, 0,
      -0.19426484441300861, 0}, {0.033405975318093875, -0.03951056796092695,
      0.19426484448105495, -2.0771956573394124*^-9, 0.033405975318093875},
     {0, 0.033405975469619195, 0, -0.033405975469619 2, 0}},
    {{-1.7590992572972645*^-6, -0.003396779100852105 -
       3.5144911803344 32*^-8*I, -1.0805575723769594*^-6,
      0.003397497174748252 - 3.5144911803344 32*^-8*I,
      -1.7590992572972645*^-6}, {0.003397917905116128 +
       3.5144911803159 88*^-8*I, -2.6061747855439034*^-7,
      0.019755897494740927 + 3.5144911800321895*^-8*I,
      -0.004018188761633151, 0.003397917905116128 + 3.5144911803159 88*^-8*
        I}, {-1.0805575723724215*^-6, -0.019755090812208102 -
       3.5144911802264 46*^-8*I, -4.3796978333326132*^-7,
      0.019755476796745795 - 3.5144911802264 46*^-8*I,
      -1.0805575723724215*^-6}, {-0.003396359139175 7628 +
       3.5144911803159 67*^-8*I, 0.004017670731205961,
      -0.019754670818151901 + 3.5144911800321895*^-8*I,
      -2.5840852176450 23*^-7, -0.003396359139175 7628 +
       3.5144911803159 67*^-8*I}, {-1.7590992572972645*^-6,
      -0.003396779100852105 - 3.5144911803344 32*^-8*I,
      -1.0805575723769594*^-6, 0.003397497174748252 - 3.5144911803344 32*^-8*
        I, -1.7590992572972645*^-6}}, {{0, 0.005744523898888 98745, 0,
      -0.005744523898888 98745, 0}, {-0.005744526484050362,
     4.916240124160756*^-9, -0.03340597616795204, 0.006794276659946141,
      -0.005744526484050362}, {0, 0.033405974619761046, 0,
      -0.033405974619761046, 0}, {0.005744526484050362,
      -0.006794276659946141, 0.03340597616795204, -4.916240124160756*^-9,
      0.005744526484050362}, {0, 0.005744523898888 98745, 0,
      -0.005744523898888 98745, 0}}}},
   {{{{0, -0.005744546691143481, 0, 0.005744546691143478, 0},
     {9.735818095504373*^-6, 0.003392839872638108 - 1.9451118908416385*^-7*
        I, 5.9803966134780 02*^-6, -0.003401466933934258 -
       1.9451118976268978*^-7*I, 9.735818095504373*^-6},
     {0, -0.033406032366685 52, 0, 0.033406032366685524, 0},
     {-9.735818095504 93*^-6, 0.003401466933934257 + 1.9451118976107084*^-7*
        I, -5.9803966134778 94*^-6, -0.003392839872638 1075 +
       1.9451118908369 35*^-7*I, -9.735818095504 93*^-6},
     {0, -0.005744546691143481, 0, 0.005744546691143478, 0}},
    {{0.005744532383 4868795, 2.7209176589777784*^-8, 0.033406028501718754,
      -0.006794304679387975, 0.005744532383 4868795},
     {-0.003399138381560 3535 + 1.9451118903697218*^-7*I,
      1.4301733411163815*^-6, -0.019756417205627878 + 1.9451118821979373*^-7*
        I, 0.004019386383227855, -0.003399138381560 3535 +
       1.9451118903697218*^-7*I}, {0.03340603320529245,
     1.1496321249 13813*^-8, 0.19426502495 03809, -0.039510700111156743,
      0.03340603320529245}, {-0.003395164170659298 - 1.9451118903500796*^-7*
        I, -1.4423989081599128*^-6, -0.019754280957921 76 -
       1.9451118821782964*^-7*I, 0.004016519321050392,
```

```
    -0.003395164170659298 - 1.9451118903500796*^-7*I},
   {0.0057445323834868795, 2.7209176589777784*^-8, 0.033406028501718754,
    -0.006794304679387975, 0.0057445323834868795}},
  {{0, -0.033406037070265654, 0, 0.033406037070265654, 0},
   {5.980396553317465*^-6, 0.01975195648074519 - 1.9451119015145247*^-7*I,
    2.4239635644669265*^-6, -0.01975874557882411 - 1.9451118947299096*^-7*
     I, 5.980396553317465*^-6}, {0, -0.19426502871643148, 0,
    0.19426502871643148, 0}, {-5.9803965533169056*^-6,
    0.019758745578824108 + 1.9451118947137202*^-7*I,
    -2.423963564467034*^-6, -0.019751956480745188 + 1.9451119014881372*^-7*
     I, -5.9803965533169056*^-6}, {0, -0.033406037070265654, 0,
    0.033406037070265654, 0}}, {{-0.0057445323834868795,
    0.006794304679387974, -0.03340602850171875, -2.720917658804306*^-8,
    -0.0057445323834868795}, {0.003395164170659297 +
     1.9451118903807629*^-7*I, -0.0040165193210503925,
    0.019754280957921755 + 1.9451118821979362*^-7*I,
    1.4423989081586118*^-6, 0.003395164170659297 + 1.9451118903807629*^-7*
     I}, {-0.03340603320529245, 0.039510700111567425,
    -0.1942650249503809, -1.1496321256077024*^-8, -0.03340603320529245},
   {0.003399138381560354 - 1.945111890361127*^-7*I,
    -0.0040193863832278555, 0.019756417205627885 - 1.9451118821782975*^-7*
     I, -1.4301733411142131*^-6, 0.003399138381560354 -
     1.945111890361127*^-7*I}, {-0.0057445323834868795,
    0.006794304679387974, -0.03340602850171875, -2.720917658804306*^-8,
    -0.0057445323834868795}}, {{0, -0.005744546691143481, 0,
    0.0057445466911434478, 0}, {9.735818095504373*^-6,
    0.003392839872638108 - 1.9451118908416385*^-7*I,
    5.980396613478002*^-6, -0.0034014669333934258 - 1.9451118976268978*^-7*
     I, 9.735818095504373*^-6}, {0, -0.03340603236668552, 0,
    0.03340603236668524, 0}, {-9.735818095504493*^-6,
    0.0034014669333934257 + 1.9451118976107084*^-7*I,
    -5.980396613477894*^-6, -0.0033928398726381075 +
     1.945111890836935*^-7*I, -9.7358180955049 3*^-6},
   {0, -0.005744546691143481, 0, 0.005744546691143478, 0}}},
 {{{0, -0.0057445374411731 66, 0, 0.0057445374411731 67, 0},
   {6.909111431334132*^-6, 0.003394084952960 8998 - 1.380366212983 14*^-7*I,
    4.244042776431861*^-6, -0.003400207226271937 - 1.380366215595 0352*^-7*
     I, 6.909111431334132*^-6}, {0, -0.03340600379234614, 0,
    0.03340600379234614, 0}, {-6.909111431333901*^-6,
    0.003400207226271 9367 + 1.380366215600 6738*^-7*I,
    -4.244042776431004*^-6, -0.003394084952960 9006 +
     1.380366212999725*^-7*I, -6.909111431333901*^-6},
   {0, -0.005744537441173166, 0, 0.005744537441173167, 0}},
  {{0.005744527287602 46, 1.9309234111528479*^-8, 0.03340600104954062,
    -0.006794290669658324, 0.005744527287602 46},
   {-0.003398554748 10932 + 1.380366214703945 1*^-7*I,
    1.0149358463471675*^-6, -0.019756074331108636 + 1.380366210859121 3*^-7*
     I, 0.004018958617976579, -0.003398554748 10932 +
     1.380366214703945 1*^-7*I}, {0.033406004387 4675,
    8.158474555652884*^-9, 0.19426493431552616, -0.039510634036217 05,
    0.033406004387 47675}, {-0.003395734411980 6727 - 1.380366214685619 9*^-7*
     I, -1.0236118348988868*^-6, -0.019754558322681696 -
     1.380366210840787 5*^-7*I, 0.004016923980559744,
    -0.003395734411980 6727 - 1.380366214685619 9*^-7*I},
   {0.005744527287602 46, 1.9309234115284 79*^-8, 0.033406001049540 62,
    -0.006794290669658324, 0.005744527287602 46}},
  {{0, -0.033406007130 28473, 0, 0.033406007130 28473, 0},
   {4.244042755299643 5*^-6, 0.019752908736521198 - 1.380366214709194*^-7*I,
    1.7201881085810272*^-6, -0.019757726682116474 - 1.380366212095674*^-7*
     I, 4.244042755299643 5*^-6}, {0, -0.1942649369881408, 0,
    0.1942649369881408, 0}, {-4.244042755299873*^-6,
    0.019757726682116478 + 1.380366212105206*^-7*I,
    -1.720188108581885*^-6, -0.019752908736521198 + 1.380366214704090 7*^-7*
     I, -4.244042755299873*^-6}, {0, -0.033406007130 28473, 0,
    0.033406007130 28473, 0}}, {{-0.005744527287602 46,
    0.006794290669658324, -0.033406001049540626, -1.930923411528479*^-8,
    -0.005744527287602 46}, {0.003395734411980 6727 + 1.380366214703943*^-7*
     I, -0.004016923980559743, 0.019754558322681 69 +
     1.380366210947471*^-7*I, 1.0236118348967184*^-6,
    0.003395734411980 6727 + 1.380366214703943*^-7*I},
   {-0.033406004387 47674, 0.039510634036217 05, -0.19426493431552616,
    -8.158474555652884*^-9, -0.033406004387 47674},
   {0.003398554748 10932 - 1.380366214685622*^-7*I, -0.0040189586179765775,
    0.019756074331108 64 - 1.380366210929135 1*^-7*I,
    -1.0149358463480349*^-6, 0.003398554748 10932 - 1.380366214685622*^-7*
     I}, {-0.005744527287602 46, 0.006794290669658324,
    -0.033406001049540626, -1.930923411528479*^-8, -0.005744527287602 46}},
  {{0, -0.005744537441173166, 0, 0.005744537441173167, 0},
   {6.909111431334132*^-6, 0.003394084952960 8998 - 1.380366212983 14*^-7*I,
    4.244042776431861*^-6, -0.003400207226271937 - 1.380366215595 0352*^-7*
     I, 6.909111431334132*^-6}, {0, -0.03340600379234614, 0,
    0.03340600379234614, 0}, {-6.909111431333901*^-6,
    0.003400207226271 9367 + 1.380366215600 6738*^-7*I,
    -4.244042776431004*^-6, -0.003394084952960 9006 +
     1.380366212999725*^-7*I, -6.909111431333901*^-6},
   {0, -0.005744537441173166, 0, 0.005744537441173167, 0}}},
 {{{0, -0.005744526484049746, 0, 0.005744526484049744, 0},
   {1.7590992552652045*^-6, 0.003396359139175697 7 - 3.514491182394117*^-8*
     I, 1.080557573865912*^-6, -0.0033979179051160926 -
```

```
      3.5144911856348764*^-8*I, 1.7590992552652045*^-6},
     {0, -0.033405975318093806, 0, 0.033405975318093833, 0},
     {-1.7590992552658855*^-6, 0.003397917905116092 + 3.5144911858980317*^-8*
        I, -1.0805575738746633*^-6, -0.003396359139175697 +
      3.5144911822215822*^-8*I, -1.7590992552658855*^-6},
     {0, -0.005744526484049746, 0, 0.005744526484049744, 0}},
    {{0.005744523898890889, 4.9162407651792694*^-9, 0.03340597461976025,
      -0.006794276659945634, 0.005744523898890889},
     {-0.003397497174746668 + 3.514491185273212*^-8*I,
      2.584085221296953*^-7, -0.019755476796745726 + 3.5144911806272524*^-8*
        I, 0.004018188761632 84, -0.003397497174746668 +
      3.5144911852732 12*^-8*I}, {0.033405975469618196,
      2.0771950132170147*^-9, 0.19426484413008688, -0.03951056796092742,
      0.033405975469618196}, {-0.003396779100853683 - 3.5144911853632164*^-8*
        I, -2.6061747894533596*^-7, -0.0197550908122082 -
      3.5144911803625417*^-8*I, 0.004017670731206252,
     -0.003396779100853683 - 3.5144911853632164*^-8*I},
     {0.005744523898890889, 4.9162407651792694*^-9, 0.03340597461976025,
      -0.006794276659945634, 0.005744523898890889}},
    {{0, -0.033405976167952066, 0, 0.033405976167952066, 0},
     {1.0805575711450157*^-6, 0.019754670818158723 - 3.5144911834349 51*^-8*
        I, 4.3796978362874 38*^-7, -0.019755897494741374 -
      3.5144911802138655*^-8*I, 1.0805575711450157*^-6},
     {0, -0.1942648448105502, 0, 0.19426484481055017, 0},
     {-1.0805575711443 35*^-6, 0.019755897494741374 + 3.5144911799999 72*^-8*
        I, -4.3796978362002304*^-7, -0.019754670818158723 +
      3.5144911836527286*^-8*I, -1.0805575711443 35*^-6},
     {0, -0.033405976167952066, 0, 0.033405976167952066, 0}},
    {{-0.005744523898890889, 0.006794276659945633, -0.03340597461976026,
      -4.916240764311908*^-9, -0.005744523898890889},
     {0.0033967791008536826 + 3.514491185383633*^-8*I,
      -0.004017670731206255, 0.019755090812208193 + 3.5144911802725186*^-8*
        I, 2.6061747894316756*^-7, 0.0033967791008536826 +
      3.5144911185383633*^-8*I}, {-0.03340597546961818, 0.03951056796092742,
      -0.19426484413008688, -2.07719501321701 47*^-9, -0.03340597546961818},
     {0.0033974971747466687 - 3.5144911854736664*^-8*I,
      -0.004018188761632837, 0.01975547679674573 - 3.5144911803625 47*^-8*I,
      -2.584085221270932*^-7, 0.0033974971747466687 - 3.5144911854736 64*^-8*
        I}, {-0.005744523898890889, 0.006794276659945633,
      -0.03340597461976026, -4.916240764311908*^-9, -0.005744523898890889}},
     {{0, -0.005744526484049746, 0, 0.005744526484049744, 0},
      {1.7590992552652045*^-6, 0.0033963591391756977 - 3.5144911823941 17*^-8*
        I, 1.0805575738659 12*^-6, -0.003397917905116092 -
       3.5144911856348764*^-8*I, 1.7590992552652045*^-6},
      {0, -0.033405975318093806, 0, 0.033405975318093833, 0},
      {-1.7590992552658855*^-6, 0.003397917905116092 + 3.5144911858980317*^-8*
        I, -1.0805575738746633*^-6, -0.003396359139175697 +
       3.5144911822215822*^-8*I, -1.7590992552658855*^-6},
      {0, -0.005744526484049746, 0, 0.005744526484049744, 0}}}},
   {{{{0, 9.735818017592754*^-6, 0, -9.735818017593764*^-6, 0},
      {0.005744533238379583, -0.0033991383815749716 + 1.94511189227279 78*^-7*
         I, 0.033406033205262 65, -0.0033951641706343482 -
       1.9451111885512225*^-7*I, 0.005744532383479583},
      {0, 5.980396603545911*^-6, 0, -5.980396603540217*^-6, 0},
      {-0.005744532383479583, 0.003395164170634348 + 1.9451118855109338*^-7*
         I, -0.033406033205262 65, 0.003399138381574971 -
       1.9451118922725326*^-7*I, -0.005744532383479583},
      {0, 9.735818017592754*^-6, 0, -9.735818017593764*^-6, 0}},
     {{-0.005744546691152506, 0.0033928398726486998 - 1.945111876729495*^-7*
         I, -0.033406032366676651, 0.003401466933923646 +
       1.9451111883491252*^-7*I, -0.005745456691152506},
      {2.7209175156896193*^-8, 1.4301733332247174*^-6, 1.14963226507946 99*^-8,
       -1.4423988992377963*^-6, 2.7209175156896193*^-8},
      {-0.033406037070261 64, 0.019751956480725 79 - 1.9451118957161024*^-7*I,
       -0.19426502871643547, 0.019758745578843447 + 1.9451118889551617*^-7*I,
       -0.033406037070261 64}, {0.006794304679383288, -0.0040165193210249675,
       0.039510700111572095, -0.00401938638325325, 0.006794304679383288},
      {-0.005744546691152506, 0.0033928398726486998 - 1.945111876729495*^-7*
         I, -0.033406032366676651, 0.003401466933923646 +
       1.9451111883491252*^-7*I, -0.005745456691152506}},
     {{0, 5.980396595435075*^-6, 0, -5.980396595435747*^-6, 0},
      {0.033406028501737 15, -0.019756417205638345 + 1.945111900586 46*^-7*I,
       0.19426502495039952, -0.019754280957921626 - 1.9451119073480568*^-7*I,
       0.033406028501737 15}, {0, 2.4239635754637507*^-6, 0,
       -2.4239635754677605*^-6, 0}, {-0.033406028501737 15,
       0.019754280957921 63 + 1.9451119073467656*^-7*I, -0.19426502495039955,
       0.019756417205638 34 - 1.9451119005861949*^-7*I, -0.033406028501737 15},
      {0, 5.980396595435075*^-6, 0, -5.980396595435747*^-6, 0}},
     {{0.005744546691152506, -0.0034014669339236 49 - 1.945111883490842 3*^-7*
         I, 0.033406032366676504, -0.003392839872648702 +
       1.9451111876729911*^-7*I, 0.005744546691152506},
      {-0.006794304679383288, 0.004019386383253253, -0.03951070011157209,
       0.004016519321024 97, -0.006794304679383288}, {0.033406037070261 64,
       -0.019758745578843 45 - 1.9451118889547509*^-7*I, 0.19426502871643547,
       -0.019751956480725787 + 1.9451118957165143*^-7*I,
       0.033406037070261 64}, {-2.7209175158630916*^-8, 1.4423988992356 28*^-6,
       -1.1496322643855805*^-8, -1.4301733322497762*^-6,
       -2.7209175158630916*^-8}, {0.005744546691152506,
       -0.003401466933923649 - 1.9451111883490842 3*^-7*I,
```

```
    0.033406032366676504, -0.0033928398726487 02 + 1.945111876729911*^-7*I,
    0.005744546691152506}}, {{0, 9.735818017592754*^-6, 0,
    -9.735818017593764*^-6, 0}, {0.005744532383479583,
    -0.0033991383815749716 + 1.9451118922727978*^-7*I,
    0.033406033205262 65, -0.0033951641706343482 - 1.945111885512225*^-7*I,
    0.005744532383479583}, {0, 5.980396603545911*^-6, 0,
    -5.980396603540217*^-6, 0}, {-0.005744532383479583,
    0.0033951641706343 48 + 1.9451118855109338*^-7*I, -0.033406033205262 65,
    0.003399138381574971 - 1.9451118922725326*^-7*I,
    -0.005744532383479583}, {0, 9.735818017592754*^-6, 0,
    -9.735818017593764*^-6, 0}}},
  {{{0, 6.9091114107490915*^-6, 0, -6.909111410747895*^-6, 0},
    {0.005744527287599649, -0.00339855474811634 + 1.3803662143495482*^-7*I,
     0.033406004387465264, -0.0033957344119696 88 - 1.3803662117515856*^-7*
       I, 0.005744527287599649}, {0, 4.244042768948524*^-6, 0,
     -4.244042768947355*^-6, 0}, {-0.005744527287599651,
     0.0033957344119696885 + 1.3803662117503802*^-7*I,
     -0.033406004387465 27, 0.003398554748116338 - 1.3803662143479144*^-7*I,
     -0.005744527287599651}, {0, 6.9091114107490915*^-6, 0,
     -6.909111410747895*^-6, 0}}, {{-0.005744537441176652,
     0.003394084952967017 - 1.3803662076206258*^-7*I, -0.033406003792342 66,
     0.003400207226 2581 + 1.3803662102229884*^-7*I,
     -0.005744537441176652}, {1.9309233567112172*^-8,
     1.0149358428625417*^-6, 8.158475096886608*^-9, -1.023611831355714*^-6,
     1.9309233567112172*^-8}, {-0.033406007130283 19,
     0.019752908736516056 - 1.380366213168904*^-7*I, -0.19426493698814232,
     0.019757726682121578 + 1.3803662105719 91*^-7*I, -0.033406007130283 19},
    {0.0067942906696565105, -0.004016923980550063, 0.039510634036218836,
     -0.004018958617986247, 0.0067942906696565105},
    {-0.005744537441176652, 0.003394084952967017 - 1.3803662076206258*^-7*
       I, -0.033406003792342 66, 0.003400207226 26581 +
      1.3803662102229884*^-7*I, -0.005744537441176652}},
   {{0, 4.244042766823742*^-6, 0, -4.244042766820049*^-6, 0},
    {0.033406001049547 68, -0.019756074331111 83 + 1.3803662150737952*^-7*I,
     0.19426493431553335, -0.019754558322682452 - 1.3803662176713295*^-7*I,
     0.033406001049547 68}, {0, 1.720188110193226*^-6, 0,
     -1.720188110199284*^-6, 0}, {-0.033406001049547 68,
     0.019754558322682456 + 1.380366217670124*^-7*I, -0.19426493431553335,
     0.019756074331111 83 - 1.3803662150764983*^-7*I, -0.033406001049547 68},
    {0, 4.244042766823742*^-6, 0, -4.244042766820049*^-6, 0}},
   {{0.005744537441176652, -0.003400207226 2658144 - 1.3803662102202652*^-7*
       I, 0.033406003792342 67, -0.003394084952967016 +
      1.3803662076233551*^-7*I, 0.005744537441176652},
    {-0.0067942906696565105, 0.004018958617986247, -0.039510634036218815,
     0.004016923980550068, -0.0067942906696565105},
    {0.033406007130283175, -0.019757726682121 58 - 1.3803662105692666*^-7*I,
     0.19426493698814232, -0.019752908736516053 + 1.3803662131716263*^-7*I,
     0.033406007130283175}, {-1.9309233566244 81*^-8, 1.0236118313518 11*^-6,
     -8.158475089947714*^-9, -1.014935842864 7101*^-6,
     -1.9309233566244 81*^-8}, {0.005744537441176652,
     -0.003400207226 2658144 - 1.3803662102202652*^-7*I,
     0.033406003792342 67, -0.003394084952967016 + 1.3803662076233551*^-7*I,
     0.005744537441176652}}, {{0, 6.9091114107490915*^-6, 0,
     -6.909111410747895*^-6, 0}, {0.005744527287599649,
     -0.003398554748116 34 + 1.3803662143495482*^-7*I, 0.033406004387465264,
     -0.003395734411969688 - 1.3803662117515856*^-7*I,
     0.005744527287599649}, {0, 4.244042768948524*^-6, 0,
     -4.244042768947355*^-6, 0}, {-0.005744527287599651,
     0.0033957344119696885 + 1.3803662117503802*^-7*I,
     -0.033406004387465 27, 0.003398554748116338 - 1.3803662143479144*^-7*I,
     -0.005744527287599651}, {0, 6.9091114107490915*^-6, 0,
     -6.909111410747895*^-6, 0}}},
  {{{0, 1.7590992531274244*^-6, 0, -1.7590992531286808*^-6, 0},
    {0.005744523898890513, -0.003397497174747721 + 3.5144911848724184*^-8*
       I, 0.033405975469616635, -0.003396779100852104 -
      3.5144911813275 02*^-8*I, 0.005744523898890513},
    {0, 1.0805575726026522*^-6, 0, -1.080557572602192*^-6, 0},
    {-0.005744523898890514, 0.003396779100852104 + 3.5144911812819774*^-8*
       I, -0.033405975469616614, 0.003397497174774772 -
      3.5144911848277 48*^-8*I, -0.005744523898890514},
    {0, 1.7590992531274244*^-6, 0, -1.7590992531286808*^-6, 0}},
   {{-0.005744526484050 2355, 0.003396359139176 6696 - 3.5144911762015 65*^-8*
       I, -0.033405975318093 35, 0.003397917905115116 +
      3.5144911797398 4*^-8*I, -0.005744526484050 2355},
    {4.916240696657 692*^-9, 2.584085216647894*^-7, 2.07719507566706*^-9,
     -2.606174784838363*^-7, 4.916240696657 692*^-9},
    {-0.033405975161795185, 0.019754670818158 17 - 3.5144911818287 98*^-8*I,
     -0.19426484481055037, 0.019755897494741888 + 3.5144911783091 7*^-8*I,
     -0.033405975161795185}, {0.006794276659945 3875, -0.004017670731204 9465,
     0.039510567960927 64, -0.004018188761634 1444, 0.006794276659945 3875},
    {-0.005744526484050 2355, 0.003396359139176 6696 - 3.5144911762015 65*^-8*
       I, -0.033405975318093 35, 0.003397917905115116 +
      3.5144911797398 4*^-8*I, -0.005744526484050 2355}},
   {{0, 1.0805575723771528*^-6, 0, -1.080557572377735*^-6, 0},
    {0.033405974619761 2, -0.019755476796746 108 + 3.5144911840484 25*^-8*I,
     0.19426484413008 78, -0.019755509081220835 - 3.5144911875725066*^-8*I,
     0.033405974619761 2}, {0, 4.379697836198045*^-7, 0,
     -4.379697836606021*^-7, 0}, {-0.033405974619761 2,
     0.019755509081220835 + 3.5144911871 6371 8*^-8*I, -0.19426484413008782,
```

```
       0.0197554767967461O8 - 3.5144911840904905*^-8*I, -0.0334059746197612},
      {0, 1.08055757237717528*^-6, 0, -1.080557572377735*^-6, 0}},
     {{0.00574452648840502355, -0.00339791790511511 55 - 3.514491179730511*^-8*
         I, 0.03340597531809336, -0.003396359139176669 +
        3.5144911762108944*^-8*I, 0.00574452648405023 55},
      {-0.006794276659945387 5, 0.0040181887616341444, -0.039510567960927 64,
       0.0040176707312049465, -0.006794276659945387 5},
      {0.03340597616795185, -0.019755897494741 89 - 3.514491178299842*^-8*I,
       0.19426484481055037, -0.019754670818158175 + 3.514491181838126*^-8*I,
       0.03340597616795185}, {-4.916240696657692*^-9, 2.606174784387967*^-7,
       -2.077195068728166*^-9, -2.584085216639220 3*^-7,
       -4.916240696657692*^-9}, {0.00574452648840502355,
       -0.003397917905115 5155 - 3.514491179730511*^-8*I, 0.03340597531809336,
       -0.003396359139176669 + 3.5144911762108944*^-8*I,
       0.00574452648840502355}}, {{0, 1.75909925312742 44*^-6, 0,
       -1.75909925312868 08*^-6, 0}, {0.00574452389890 513,
       -0.003397497174747721 + 3.5144911848724184*^-8*I,
       0.033405975469616635, -0.003396779100852104 - 3.514491181327502*^-8*I,
       0.00574452389890513}, {0, 1.08055757260265 22*^-6, 0,
       -1.0805557572602192*^-6, 0}, {-0.00574452389890514,
       0.003396779100852104 + 3.5144911812819774*^-8*I,
       -0.033405975469616614, 0.003397497174747772 - 3.514491184827748*^-8*I,
       -0.00574452389890514}, {0, 1.75909925312742 44*^-6, 0,
       -1.75909925312868 08*^-6, 0}}}},
   {{{{3.598735888376100111675377827861292157140828623230793786772081621874765\
770944416210772706834336735095455917534083239502456719397667096496 8‘119.87019\
782302441 - 4.062787305450733427417385230512537435443874586702672862026481307\
977955295847415996035612653648812448857827146308607006456878572965‘112.9228719\
3491131*^-7*I, -0.6826410901844929088090794113922846042649420641146997090098\
502891342100443642910482413027519320966066152853195301521647594930478602916‘\
119.63286879837787 - 1.072630085593226609613577993041953198680504718460507499\
236997171755736773281468850269465273689493247640984753349359993330761165 36‘11\
2.82912634625784*^-7*I, 2.514995166334721986703706366320202809705829938755208\
350868916293351279201441093769799773555095487024328496271540239764949494867632\
0815996‘120.12968458929437 - 3.434527108811046592049377311457495967025645723 2\
070350754319339642674588781617707164728326967314520277619219640771203335 4847\
5054‘112.2650143832123*^-8*I, -0.736702165555557370941151375485929414421799859\
110809127213116484222389581426655674800182499712648434420684565183746067968 9\
603110851219 36‘119.66405190377995 + 1.267264458117114673587537308720614600177\
198454483936078196078701820988942849731529095142117136936411895949233734270528\
787585826756‘112.899627212057 15*^-7*I, 4.155897137204208196936393973873777051\
622814568512458077826722257111042567141588060640460129999190933133713513704 42\
682880951511440811733 5‘119.90869738639329 + 1.115372664449049542240832937425 8\
123657780122702652527270180584424480933456940058051676838002484888882353681 45\
374516512334845873 27‘112.337452593214 95*^-7*I},
      {-0.6826420490394630787591126296308406460480623040062932650446892352108\
668979624777095497935099459193179801228461674357380786683426463444 27‘119.6328\
6934382994 + 9.11445404848110332895559437993884745178374069868158003816025517\
163392260997038419975079617007225541587433196462093847998477600 2‘112.75840696\
734613*^-8*I, 0.4706537957382305773373088052207030296083953757752211806767322\
752882476477594538104893238405367582071010975433664065708728409515171859 97‘11\
9.8367651610188 + 1.567893313672874384122538627601357176867301743170888144008\
750425472168192277662452660673090988757540511722121172668958166350296‘112.35\
93801035291*^-8*I, -2.315026313824762865246208156546440182026029969105596842\
572955428895772242452463709834267720578158571945062497967715756079212792679768\
026‘120.1432520563948 4 - 0.215337336126088883515893526816641252377502152127 63\
845380062938219941405283781133596993499219181050954114031819955066615506013 23\
25772776‘119.111815460812 62*I, 0.43974682066418142184198410707036498730633191\
765885838453879656765562697630986183437502101122511190305181005015100485 13071\
395037017851 15‘119.842094029205 + 0.007264466008417320088058932058887696684 52\
945318144078198772831389473392769400246381842854537079904835772581635164 17983\
668909094758350 13‘118.060095016954 55*I, -0.607408257964641665281365653284 175\
201531368073181015544673237542930580834909685360965991987285549221003491 71228\
163998832303907294679393‘119.593650122603 24 - 0.026588762784468073603792 87088\
517378880549141557180967772440891894298444433804607376514692550687 75367798230\
532854052150489157462036357 21‘118.234867559988 67*I},
      {2.514995329285328952343645352410219677588044303666476391429021076 28281\
357577828146316825840938587955751067844252692480376926829681043181 9‘120.12968\
458480158 - 2.730186518979951295732245260085828871432378529873914023412 42093\
104558573678593091602104987836926239031396954285724643412342772 5‘112.16533971\
983938*^-8*I, 0.18634571769657240707581848964712710427719996688064288582 84921\
714994231859338872530204303887814664552925917596388753999582168645986 26817‘11\
9.682906192014 24 + 0.215337315378079636471397557963134972863510182054821 84721\
009443381944840584484102465730131512700024195237349244105666487960165066380 48\
5‘119.745706069458 56*I, -0.129838186818138992363950529606714005611666637321 534\
742769559588502506692745610692818022118997015803689822813185335701082500 81793\
2492445‘120.14766044744225 - 2.186340231576519324212387054976043037460987 9394\
054617995590476222176508876728569688558077554806519295503313388295484090 7394\
601‘112.37397575192712*^-9*I, 0.217460297467140290622110119571119143106178 376\
791804725638371450712277974165602753428169995039787963235612337161115564 79353\
0133255068061‘119.74227448140866 + 0.207122840644611192789705234424344234 12303\
013487069398594132898494038420144922692859717632392042323934597151454300 50058\
301839755827783029‘119.721122497356 42*I, 2.616961332192118567708200016428 9876\
292402379182289475985158570959234039225274345237160269758610720265211791 83681\
754181935965160394374 05‘120.12818167963985 + 2.806813547943560700076204952 348\
284444936018391567285203948635197061364940224974582277439360558939753801 43455\
257805753170595195 49‘112.15859793813623*^-8*I},
      {-0.736701414918218747030071022895342742763881773920464529222428912 5912\
639611540964325738157294793986523171352442365261407438550364637882 76‘119.6640\
```

```
5149926902 - 8.2941625480073789021543456591608348525452323535617455221433039\
9547886207798063365371686433483546161312584041143747803901863884‘112.7155325\
66459⋆^-8⋆I, 0.48347773576682084276712188980495546731157060775513562827997632\
2837946835620711060744485213821074763744194336527792699192929091689193876‘119\
.88100823359478 - 0.007264460322160707063030831120015226471985988678014381377\
1404557076977332867536837377522886708768721091117003538230176964058115713987 5‘\
118.05783550008182⋆I, -2.3339463364645197334294036281815016721268756903784148\
9125177195889741411297374639184562732022296441798194492505047788380776938882\
525824‘120.14402862296417 - 0.207122815076826131233130175576950323363175924 05\
9865167131594646123566696699700661199447406670048840285442921930876653806095 3\
49860201548‘119.092165696818 31⋆I, 0.46241214798501875072718309935406661002558\
6321216122513993015591404492805428481402572686324358739907175504104936689759 07\
1990638541153536 9‘119.82055958934033 - 2.4737209568978863759870431051259819 58\
8787535816235239112499115567572383455451687070381228366878271362163124423734 3\
0258218311778 74‘112.548881062812 55⋆^-8⋆I, -0.8308761043752862348431423605871 31\
0658308856792635268153219075443843412287349731522469525832048661510571255678 1\
7453400764826925850277329‘119.71636868212029 + 0.026588755390263016617612656 4\
7664315538716500076854288620827642578414903577195013033818935355138533759067 7\
6131437936408628756899645596 63‘118.22153042173109⋆I},
      {4.155899404675120742490169491853687100626447543645117669826336396846 52\
5001486593767729329289765096113430786051088144599012510914630174125‘119.90869\
751427421 + 1.526071892913206291627909693754923323885783900188794151268601445\
0746465271593409735518897477501844972348364072283462044950146556 2‘112.4736074\
8116466⋆^-7⋆I, -0.8308760531930588844547532488774890830859348634553021142 51\
4577974769307932895065205638445061217130993319414363597226380202916473833081‘\
119.7163686969217 + 0.026588672047733569565232013835515450120036161329789 81\
4762472549551870646194280990150781643960894872585925685269254387932904000016\
342‘118.221529101985 75⋆I, 2.6169616884447196577709801745392452178439699048750\
1670611648872363069733502654631983419379718774717753065040655900607212467 26\
542774794‘120.128181677897 94 + 3.406604392578271101258506376767057311189284340\
1118951164950423934817232191825286955229212121827820235690931948107891293886 0\
807723‘112.24270601523105⋆^-8⋆I, -0.607408523806852753296765507653743200226 77\
7793614731897631324704056080166663177698420696276646954078946797165013653 8854\
5027024861013422 8‘119.593650299950 18 - 0.026588695826643271845872557431415638\
5899727185213351855447879477645756067055127676173594252501770782599615191303 8\
8921176310219902205585‘118.23486645358508⋆I, 4.155895360882637725820009049707\
7299020140748205441470712135755916738396310900102876316263488245217235249860 2\
3005140945832073643171320 24‘119.908697287864 51 + 1.526071399661751018984532 09\
2419744277141858862614240346152650749106489262306486783634827260194480429583 4\
1349463721981952483079 57‘112.47360753696329⋆^-7⋆I}},
     {{-0.6826421485087161827197098216474666348881144542822628445617376469908\
1031240072117192881544998302084653528625619507623052707892492694916 5‘119.6328\
693875421 + 9.933631317465629550115876670970959002739343343442767771654094487\
8212145688438361889757526696331669410574688300803832242379324548‘112.79578432\
534373⋆^-8⋆I, 0.4706538889770054636586668760810178598140246943047077798733653\
1477806113272900574048247078224946152774477601615488683082973238685268532 6‘11\
9.836765521718 93 + 1.2963437442336267974247911974789526918143842443559356341 8\
4134555637144472994532468974284379842175566900173995107537163188590330 9‘112.2\
7678404638665⋆^-8⋆I, 0.1863456523973721001175733612182776206256499708316401 25\
5047601373554277258725507858176843699966160451174124798518073698798714737272\
4545‘119.68290609223858 + 0.2153373353508194989208244071951712072318046669003\
3978519146076252036555687617661158203111214063053059462044914679011260666837 3\
495954116‘119.74570616214945⋆I, 0.4834778067555169494950443132205936415698725\
2280795372092616998644158517161376624619418585987058486065398456431806841314 5\
8965554985469 37‘119.881008425136 97 - 0.007264491472548166908719583369661989 3\
8811456562147310546901050665414250351936208971730367976416770863294808533847 2\
2200452602744543891 6‘118.057837100130 41⋆I, -0.8308755814561085992167785164057\
7601043208688745271465573015658357772215581089656498893674643004202577315516 6\
9405367776079334328974721 53‘119.71636847429109 + 0.026588613281687465722344 27\
8561304362100431615000398925343016402261673232835080566659910320313276587548 1\
1703943364786751767925164823334 5‘118.22152816605474⋆I},
      {0.4706540208108545700939497451928878275454622457932948126416119054604 2\
2193836856661335812881849312248789961687960756461363918084868320699‘119.83676\
550062575 - 4.0960832792035602785082497441712773813056724885358461523785174 09\
6660806955983447066769691713076824032041744273041993025513829264‘112.77643250\
527812⋆^-8⋆I, 0.0520280946225420041892275815607360911415872204207182469425618\
7441837789859884732690794343626950160170964861277894996706062478260553256‘11\
9.343605133776954 - 1.1015345629074862204683896494398496619466796190805158674 2\
7234811514713916135590873971184861984001509448216815897048680166682966‘112.6\
6936534493422⋆^-8⋆I, 0.1663544860635917572120844956303792333321692288353076312\
4167143048328832330766429455052800521902150204482941636422546439699445529548 14\
1067‘119.93143741004288 + 6.1593200859873678279090545211373439425352386537455\
1125573406996641079308187991722348946682809440931636233681662127376073556936 06‘\
112.49993566708983⋆^-9⋆I, 0.1056144729707583081775780461915413102435322639008\
7476390804687045492594128988474362573439334405093816068434131939813567335722 1\
15599416‘119.66911696862387 + 2.9906840718769885751993422946362248110647733235\
0663473461101987330124821849112303698788663117418243173751895810661998276185 1\
3125‘112.12116407013713⋆^-9⋆I, 0.4624120717964181390212876459454504596861547\
5771267578563390218872123246039069079699896837697090130555904866207521163221 74\
9107085728895 3‘119.82055963487956 + 3.3207402949284013984674536255911301254 35\
6269251420813766206204427359962809006967768393020323994200458111845922917385 3\
63370103392‘112.67676538363091⋆^-8⋆I},
      {-2.3150264716820014170606226431071758965183092905618680798825934501878\
4923322160162384100584793444147311535472557904385669256183120941666 4‘120.1432\
5205889897 - 0.2153373178271227420836368912825690915616551803506028243916871 8\
1975847009499272241473052978474980147671252251187465355337771312851806082‘119\
.11181539679752⋆I, 0.1663544267554127795841986934923120822010507318865125600 5\
8949518560281253511221408759823284458053058692527284814270451020952342061920 3\
56‘119.93143738616044 + 5.6256609112749350440193515676876585193459876251526 67\
```

```
914270713750011374803466692924597016618098082972701599688846966770916530511\
1.460576575421102*^-10*I, -0.4447120314223520074258291804584324349729221820068\
3113572507743442492816368150252332083445460921563814563015143251460266845807\
3312362191`120.1505095025467 - 2.221862030902410439496222712864366517342681837\
5076346929861912699072171749413022658251290234691018494314720633156416354540\
74`110.849147710780134^-10*I, 0.16848018800524412259539901715377899915632101 7\
4964523253402633427640466659865323658550240326656075986320116978191028411 3809\
6181482017488`119.95871000609154 - 1.8001468088931798686788270110462942370497\
6159970823408499073227661191010989306811468215646965561108719060943190195154 5\
20111079`110.98746909255762*^-10*I, -2.333946277389652329490079653156813414603\
2688098051181825925477283498356195151157681792178938913780459687256882150429 3\
8095740629897993153 16`120.144028621842 76 - 0.207122835519514915503296559294828\
1454274005102285548672259189701721213464409985131709537109780757712547104098 1\
3628354186570404937286058`119.09216574955357*I},
      {0.43974679606992031479995377404329082043481466488082159869306964178435\
9009925511195175181830691200163208140058606032412376341438651879656`119.84209\
391589869 + 0.0072644814621984880210380997956633606273910066015047721014062 92\
4029949082031178084098618130350273197285208852393124829248139757329565 86`118.\
06009585181617*I, 0.10561451911693734498968983619208166375655734654661033792 8\
5377887033289443227141120495020522315945166329706645396755884605550777758295 63\
6`119.66911715693426 + 3.338367752286971163454148728260551954094152748259319 9\
0601427156797157458743402779608868080366069052406395082299015013172120198`11\
2.16892770754033*^-9*I, 0.16848012109790087910354239787418088439046392219475 7\
6700472559749895148831462124998844757475220828773707342896335132085530855476 6\
8642735`119.95870998959502 - 5.11368448644286701907965009012711038642991317 8\
0504696973357249910674117442010597073095673574326949940271510437443949209410 5\
786`112.44089525221034*^-9*I, 0.10561443098107596715343051027415407575404797 3\
6245763411971319180808596511627766413066568727850442604976298353195544505250 7\
8641918948859`119.66911682907786 + 3.338366455430810939571310298873471357142 4\
2354567909408533052717239896597823346075602850577544691474537956639535364330 6\
07077634`112.1689275733946*^-9*I, 0.48347779877798703279028665917423303279020 3\
3023179019570213037371151355946173320759541511174112331385480369193991583832 1\
1162969097424864 3`119.88100854327097 - 0.007264473303270252930552561873680748 5\
5672335787003915616928740502179060387011838875326655609612763439046960765881 9\
6577848991471442215829`118.05783613921162*I},
      {-0.607408995771524565882199337689387178699103941990090353423866661295\
6324442047051874980213603059570736281930079043895932671889202017882 6`119.5936\
5067486807 - 0.026588620022406691704530552492550655573797588808089236666021702\
5921855830832164946033262328351895490045619057189810489712181479786360374`118\
.23486525287727*I, 0.46241221307870244622094610280020927469231288109363745791\
7627054477189068479024952479252648676762306056924396484456141180377384145050 10\
27`119.82055990359284 - 1.6209414927015254234167607403795736063195766002427 40\
2708647543306161607862109497050294251416016990954822480529861294576527180915 3`\
112.365297947015 16*^-8*I, 0.21746039436871970945321459696176030432748084163 58\
6340181313381181855765373979505000330787773918778393536042543847336377207 7445\
978311134`119.74227472090216 + 0.207122812579022120688760542433699360282382187\
3892143861496214382569257754734182432151081080050326474952135890248466088 7166\
62127835836818`119.721122484477 96*I, 0.43974667310987028420701821136795157511\
5339956210394775500929432491080369483929040860833641521602046526283582628 9114\
7754701796661503565 1`119.84209360587668 + 0.007264502971101039186360887161580\
0195997149270381463711384223855025486169008009556216259388544252129211 3746355\
8190542796757736896694088`118.06009694910037*I,
       -0.736701015231880093219911672299825518851069355132328211492614256 9439\
3549073770818319403865727779060009029146962683520826308026782266966 6`119.6640\
5123187508 - 9.047990814755739640658027276756324617942366810587562638380064417\
8872966603124984307070723100569765044831217974547346963804113633 9`112.7533121\
1456297*^-8*I}}, {{2.514994892621752625433858480020937329769087578852122288666\
2657691799200112016800623789102950306888437701081777691232221361177909012602 70\
25`120.12968458497943 - 3.085099176288256025719532764202801602297950283828090\
8340862654617497425210117329462282737415787231206298588503271773297263003225 *\
112.21841660734194*^-8*I, -2.315026294512690853273940080686062739620238936916\
7379848840547771850342492954216109271458787605278498910313340399456240047 6048\
1414426827`120.14325205555282 - 0.215337359149216254205961565324456455274542 4\
4743447558884895544912244841407542322917621897677630771220519530884130486587\
331318195034159`119.111815510026 79*I, -0.12983818194253782088408986851109932\
0162909484174397412143675806809357921631575546293649996193055104361951469564 4\
5591599607070666903855`120.1476605888717 + 3.569778215291093710033528207806 1\
5670421296047940216409166648181024348899686318756555333670795487758418182323 8\
6940077607885974 6`112.5868993981306 7*^-9*I, -2.333946420128997763971589882813\
7968286949500206884453029752860043436542726346068050862453451366031220791130 3\
4779320066885892934754235 11`120.14402862432298 - 0.207122797386204087050191 85\
0447853716299249912520053218230193508245518804890485761531857976396075084577 6\
1799838172654287391140415328447 1`119.092165645551541*I,
       2.616961777968192316949614513333827249524626232271810912333718803734 76\
8362546312401854942963865399047576918204199558613131625553881187002`120.12818\
168108356 + 2.451901936119812477096010759500437113370794792472590964510299342\
5049580328067525459908265176885549315383524410786380548789344476`112.09988739\
805974*^-8*I}, {0.18634557813526735069994887966173578448850032621815057010419\
2197048176802500105941219524881954917200677317572398451453359292668391630079*\
119.682905892655526 + 0.215337353819199455430727336744203085546272725474974204\
8291407722943101970395973457919449536520660100442439078015247720625047691480 4\
3313`119.74570617288755*I, 0.166354479913077926970126962170705317874769044258\
3659741897583508355078485429263645369471645935444426653591175669783012646145 7\
5860872647`119.93143730867587 + 6.904668988749074605346311925007114549802966 5\
5463849552918178153620221990604702327308303971816654084994982199143713455825\
09808`111.5495456710442*^-10*I, -0.444712017245185825574561719268266599141338\
7734450921723878514325310777454431062361020286973799546978713094201173654250\
8872056864215743`120.15050950258033 - 8.470214592953999238281988476004941090\
9024486099309054833557026903985794694933664031618674333958782546536460986692\
```

```
893376644221'111.43032505056006*^-10*I, 0.16848011468235595364174895207917561O\
60208575160227431004552716703549112281426747837767048759088284284796100701035\
04162042409955397060881'119.9587100369599 - 5.2114621843469608738171465765359S\
77942529277713963941408768411700506880391813363518606220473268284454904711756\
1267518113328592'110.44912097826561*^-11*I, 0.21746047954824688413336290653269\
94888454513250102737535737808069070455505291308942941325053562640494751705434\
60309856415248696501421391'119.74227486279555 + 0.20712279963032754834045066O5\
02699223884529085945212894131003277856905594601235988605155375889779023598073\
44935301194430334916158546427 4'119.7211224291069*I},
      {-0.12983819773921243197558482505473444752180209839651124116928249412860\
24493369341298420332913599177555094708293798288747522186742112003528'120.1476\
605635195 + 1.37934529655966422539305549315822946448302865491442637097831622630\
37176912148401994373752198410382304344358450403845589642617157'111.1739310833\
3769*^-10*I, -0.44471202641014802922206287998429902662708506598616917955993330\
32883446834517790250202611019425918008292498543100464286911271035174452439'12\
0.1505095025799 - 2.41284665422359039151444599230327179389774879696699947214O\
32952890695483251090596684897956403633994604862387205984992805999585'110.8849\
6034996982*^-10*I, -0.30521858810010367772857038936209702663947579639264040140\
36629855172362804271698535667555925584670307642322066526680750468841014897601\
314'120.15051499783199 + 6.30219540188941044389992111017768785816470613301347\
78624561954901352600552411621141146766286572878697785449964451013029965611 8'1\
11.46539588330694*^-10*I, -0.44471202641014812969223226681718933015682431457 9\
14552178891692812092632979832535051190026854053325596090225311718604103470771\
4774986782'120.15050950257991 - 2.41284674710648170626258916667677607695171177\
37382167649636106953497969752456530053315464319654994434720584679582848079180\
15014'110.884960366688O4*^-10*I, -0.12983819773921250962287122106395173697650\
96457153448072312486199206631112080203033095881871039346386498556496203394706\
08272159596133822'120.14766056351948 + 1.37934518391750530135350366678453941155\
65928143770807153136627897478739649954627330726890098355251637078947752403739\
625648218802'111.17393104787168*^-10*I}, {0.21746047954824695037103788196398 4\
87450979782377384347996120071614365124102576555018504562367126928725966651000\
00242363988633145607635 98'119.74227486279555 + 0.20712279963032748708953858645\
91993415057035683836970548209158288076826001142312611531818501681824182668394\
29106191988889286845874213675 7'119.7211224291069*I, 0.16848011468235596267889\
72542340318042030766765442497884358901508543609694349639077719396224957393127\
96328448169285125911302995736964411'119.958710036595987 -
         5.21145932709846665964555569520041809411849071412256124649166112043255\
18124608636558594292853052895035423746617274130231960619 2'110.44912074015821*^\
-11*I, -0.44471201724518579676838096174625648014515333968662525405883111317394\
03906164933988572539187672809153537415157543848051134204379566450612'120.1505\
0950258032 - 8.47021487076843643638867831033213594701756575024118231798889890\
85892851749583557733432212108175640585193214603197704522320443 7'111.430325064\
80448*^-10*I, 0.16635447991307798380960586094592088695276341438615825772166 41\
28941292755813317871291708459662646340655831941985162196171752594391615558'11\
9.93143730867585 + 6.90466902283273416721756146197399018472314045432325801088\
00244783385393878023284164620456731255262427076165179768345186539023'111.549\
545673188*^-10*I, 0.18634557813526726250115364990190779646498019620436262683989\
98070034257330919362582433828963917601919407766809939320816344835826607724019\
5'119.68290589265527 + 0.21533735381919943691592244996470102844644581299389O1\
16967015181282816659260482362626534269355763996115979029352210094713022141492\
345469'119.74570617288758*I}, {2.61696177796819278795429350805922932524908202\
66290262310625243933175078933219144267044114935610502174237732094034297134565\
85785092503596'120.12818168108356 + 2.45190197032551210141211761268082264021 4\
67067798153857560737388985874374883358497178373636499700848150405233916093729\
92921843474'112.09988740411843*^-8*I, -2.33394642012899757736579902429888990\
10871198005290057418212137964771841785537529988842905073436755220995336243040\
65724337709628285260 93'120.14402862432298 - 0.20712279738620409486922902777O3\
84161151789642482258232175721476716677641091730671439817343024349731866264799\
51384372391513152072028914 8'119.09216564551542*I,
       -0.12983818194253783646389502135286871950602297165063065279405174189 23\
88259600995128553222484489215293021120937674105265635937133086239128'120.1476\
605888717 + 3.56977819018890845835842655227564352895197708118179948491856597\
35644836551119851753335687128150866187910281371720443588945838 7'112.586899395\
07678*^-9*I, -2.31502629451269101568924354696214936484677458906915135121858430\
98584659141560516888916757060803465413841601324096056080829841400228334597'12\
0.143252055552 81 - 0.21533735914921618487220206807013820537513919443340956031\
07019541173105386841704078656234824840094357849702374188688888023269626733605\
66'119.11181551002677*I, 2.51499489262175203502374956988432470219314504610283\
69248538745418201669584978133146120033004274445626060870625384722033514645375\
71600367'120.12968458497944 - 3.08509920321075289019801552588503393498420861\
41737831145651376382286291629718974106225481074954002226587382692706847714048\
17747'112.218416611131 87*^-8*I}},
    {{-0.73670101523188087813855301867522148252653025732147395768921464094330\
29745534904023113750321229068717139509064976769958231119907455171944'119.6640\
5123187508 - 9.04799084547949886818470477531773971019842963738766103547994361\
11877592106061816540828473734125750568402768548384475466424896551'112.7533121\
1603768*^-8*I, 0.43974667310987055501763816763931847305868698193263360493336\
42026939345695880319918466541624576065335968482342057396958730447066968911'1\
19.84209360587668 + 0.00726450297110102186273865043838575723289415792438252405\
58328786337163844896305127547299662724637713071969809881668688732739252191030\
343'118.06009694910037*I, 0.21746039436871987018712537775700545186229852395\
97760717754794299498239936967273829124396821192711380308849798803771840925888\
356086796'119.74227472090215 + 0.20712281257902210167437442401149346746611256\
65582420364501426444949871939231961496853068860045008751013993068574279708717\
01286842348581'119.721122484477 96*I, 0.46241221307870260881156490975146853052\
34132457119523801671828812756654953314249064494628152050613662561921172964842\
91171215937143813591'119.82055990359284 - 1.62094145949855032198755484743491\
43737294203041967183170829726629380799363537885916791670116408667682336576941\
75502772099531357'112.36529794776457*^-8*I, -0.60740899577152520456242966402\
```

```
592957070935295486219426548044790109114297730717656767374833261698662555058562\
52183023013511499381683818241‘119.59365067486807 - 0.026588620022406647036265 4\
26306622488431162248841897889136879039134561938853439453735363286484185391234\
71034267941704264900081697782016 1‘118.23486525287727*I},
      {0.48347779877798705530920949605029566738180017122137976498068252257833\
21107662310065550452604208190487552508398735079011272244471953668771‘119.88100\
854327098 - 0.00726447330327022420738321357837416136390261747344888395702892 3\
69434933111041668820625020544163182296836747842556767982406231007555067771‘118.\
05783613921163*I, 0.10561443098107611651924574928030820401274353235154523733 6\
16314867843332237392926194635474848808915015912575553799503748670157999011677\
3‘119.66911682907786 + 3.338366474115501789275804648434795912239319644103656 1\
98347710209494663488816243948936765438291818870399006093461863757225299605‘11\
2.16892757582534*^-9*I, 0.16848012109790106336714500560699554484213739878209 0\
98679833602779089073439408355302463368505749984584691607779570076089047538680\
2607947‘119.95870998959504 - 5.11368446125723250574654833252955543162923410 3\
81129623307797794946388744971137171851571373873712886814111959664388008809405\
654‘112.44089525007144*^-9*I, 0.10561451911693740248968934603003484930113198 4\
83647181339801572684580245716103303603180150845349603225666823037204247018410\
154647434441‘119.66911715693426 + 3.338367743828134919860321768686121239713076\
81101198910831905673610490348721570256545454418958696007401172077430551625781\
07290318‘112.16892770643399*^-9*I, 0.43974679606992060407641795769342536548103\
60925313877346696328218670678570015586443143554242683078254666341426841179348\
17085506920142536‘119.84209391589877 + 0.00726448146219851420158420193044046 1\
66915981892538116354266487050143750853892931303342070516546829786836272910541\
52748948705080115463 2‘118.06009585181619*I},
      {-2.33394627738965237617823989207177400306917805681922410217743303453 27\
45678611441150652499145728485483596865409137128159831823852149359305‘120.1440\
2862184276 - 0.20712283551951497648693328705325547227191438632438886923961548\
71007417634310049356440236685777328327978961927174304647447912141442910 62‘119\
.09216574955357*I, 0.16848018800524432744901586254398104929158486062688089381\
18535035658984286254994364261849580563809872329960607438474903067896278797886\
39‘119.95871000609154 - 1.80014688074313206633603012038080562533184822893036 6\
22279070051993419405098474894559443936604580209992363621090018345233859581‘110\
.98746910989178*^-10*I, -0.44471203142235185928623240766264582941797432676216\
26324378319538415572312803389525291537780160552584661424425849440082373807837\
31224691‘120.15050950254667 - 2.22186221563144760102104131503812828222405450 0\
82863756720221042571409998045039075823175092707902086629750849033816898631696\
79‘110.84914774688802*^-10*I, 0.16635442675541291430045752311617615449562654 8\
18667199707205953872312124369647515436843133707962020804596699122004023473784\
7825766272453‘119.93143738616047 + 5.62566088030279472392954295399485415635 19\
02221151182587149753767791759777413266109655231408614712172662067713685455282\
14655661‘111.46057657303001*^-10*I, -2.31502647168200142371683784880337680965\
29410600873946723722408218695011923371696638712979886635030088055717026301516\
69327766858216197651‘120.14325205889897 - 0.21533731782712278193109584557996 1\
62087315954104691016254224712646722067954244215882988629669736197719269090754\
73897766356848628355757 22‘119.11181539679752*I},
      {0.46241207179641827985301223828771508282782290566025296566503021781574\
49317352889827635424172423873298031631396104307860486411470144354 06‘119.82055\
963487956 + 3.32074029797465709083111078584655946036007710564984127279832963 4\
06568036414693009291014204798736435449627189081035778698025893 46‘112.67676538\
40293*^-8*I, 0.10561447297075846431545598881833033289313885020528831122678660\
97483645796378366972507442547522051397049865906991846089963311276117554 86‘119\
.66911696862385 + 2.99068404650504306976929230422212702995928031178036755596 3\
85840495278884669597023224265047693216378598924018514251168666662578 5‘112.121\
1640664527*^-9*I, 0.16635448606359191384286880849676330859950183491717223859 3\
39592131438496273421209918241440346344685607495868393185531420636601211345763 1\
1‘119.93143741004288 + 6.15932008684732465922342168943571944566109500709038 09\
42553496984696532696583151922205687054572496843612620629965907981130513891‘11\
2.499935667150 45*^-9*I, 0.05202809462254207820628140828206277830988985556616 5\
92342903868306502646986967025057289753541387350451132477605736133562083494640\
400655‘119.34360513776954 - 1.10153456478368454374230243623837127861944371175\
42070182368589832733945019870821086916609106844974002664462281055976921581102\
357‘112.66936534567394*^-8*I, 0.47065402081085502264256623593439194314345736 8\
23785488776043731303117798769615792830821182526613852475324983220203024903523\
4862512110666‘119.83676550062575 - 4.09608327953990975531313819896884955141 44\
80386120941447187364716994475988502368115324627783058807990793979930966771473\
4173581952‘112.77643250531378*^-8*I},
      {-0.83087558145610961386162083090749660285329941052769904151477526917 72\
92753384448854908948940634342158728722240717386349591438743954162466‘119.7163\
6847429109 + 0.02658861328168722614700897335744373927028964610642857753816701\
86632909075637076869913683493647107011699392179626293137127152190872657 08‘118\
.22152816605474*I, 0.48347780675551735840385807104206718488332725356131678182\
52355331202546105801004377720683907241782168412812727222137426386875637003733\
11‘119.88100842513697 - 0.00726449147254823199487388910250732648992206147872 3\
36591987965818902351130124125797768107140592490281187391421509787708245802879\
2135538‘118.05783710013041*I, 0.18634565239737216543091436615950939856378511 6\
53718848093238288030219783557312364189202370957289250940001414047215835604381\
7074802200266‘119.68290609223858 + 0.21533733535081946831454382971663875066 31\
53957789710672899316055974309592988698831991408783620984545256501997728611 04\
88500705229378822 9‘119.74570616214945*I, 0.47065388897700573285172311078339 42\
36564666019383983132707218275271600373739104602083201155596004077520642013900\
53805390986529786107586‘119.83676552171896 + 1.29634374357659782162324165167\
38267520730469766392940368646265311714589003024273263438639230794710218942740\
54330891801372549846 9‘112.27678404616657*^-8*I,
       -0.68264214850871690439474877842856573475339212879141494199416414886 25\
07073111728446245450580155613123132780505992691851548322172475063132 6‘119.6328\
693875421 + 9.93363132069693751807165681841328378826909529123173702899256114 6\
03192856452071617750301606476537296187805037061673293541067190 73‘112.79578432\
548502*^-8*I}}, {{4.15589536088263829654484966535865040405835953654240167 5025\
```

```
1290430273730359456232833641626484473232265257010400228707486610141767393575l‘\
119.90869728786451 + 1.52607140025081783232670785517892282450888662091925553l\
3139279503669023952081095883456908386582206019858810033895785567403668717 3‘11\
2.47360753713099*^-7*I, -0.607408523806852565908750741591246807749666556583988\
3792690807778051305602762337323679668765346867048663972141950450878068952024 9\
1893885‘119.59365029995017 - 0.026588695826643224117796532790353904955083787897\
0096123747675442923994512141677946375858780308200458547640156901064971154137 8\
220450189884‘118.23486645358507*I, 2.6169616884447196147772905470103355024481\
7014138867794026837880770988050709702949800069082346104780328618506395918529 0\
8100926147550339‘120.12818167789794 + 3.406604413879638525358400142852117065 1\
3450572573387576979919851210126332977954606916398978420967626852284925972980 1\
7790501767452‘112.24270601794667*^-8*I, -0.830876053193057615034017172521026 6\
4291927835765458848605692672443710052259761762757188853221267514757649414645\
4197831761579062458405‘119.71636869692168 + 0.026588672047733525320404139803 1\
1061963773923363990609039649674374836273321645804066672999602640287650997611 7\
1297416299510467122913431579‘118.22152910198571*I, 4.155899404675121706380316\
3242644200759439871679633672420571935661392443112779101997090973242519681421 1\
3278424748261059057433198117899451‘119.90869751427421 +
        1.526071893562953097638404744017228218673234228644055997988075487992 0\
863207635142143804266713468205751011958039226941159455722199 6‘112.47360748134\
956*^-7*I}, {-0.830876104375285942696230774024727363124882099978411752010532 9\
773655109669414841700364776094098408602621130176678031609924271512224575 73‘11\
9.71636866821203 + 0.026588755390263181264332209442895407031755336417035555614\
7146534059933460401510455632974266783257169469819601612322109144787006790365 9\
5‘118.2215304217311*I, 0.46241214798501890475080831544028839633609378667100 66\
460620241974566871772142035375865986266133580386831948897471291948497023249 77\
682638‘119.82055958934035 - 2.47372095047000917746276155998690478587355278302\
3688854064227569134211919980508701931704736562949817396470543301891620780285 0\
577‘112.5488810616840 7*^-8*I, -2.33394633646451936651420460929574855818547142\
6687772082194767378630437709614283225357117794150323257652320004919094698570 8\
055767452748666‘120.14402862296417 - 0.207122815076826009141300906466271652980\
3134736955326544231311824608451153868300121931157374941245264943743603984937 1\
5996755489379892431‘119.09216569681831*I, 0.48347773576682087533057378125271 1\
5131876825001415376082155314129116715479635203852592195678091087917630946144 3\
6451912418024429869155879‘119.88100862359478 - 0.007264460322160717188502993 3\
5354631505595226555619629507127245679139942707217530387172718228867247439551 2\
6354197532175729631102303168481‘118.057835500081824I,
       -0.736701414918218809863759139619609775526406609185704361598914958782 1\
7017724103308026942547335407775204743941840810940268174694259839147 5‘119.6640\
5149926902 - 8.294162531154168373178939799891266883120887718885879808019371 01\
0064497563761166395540220016903430875994521548844885851015546089101‘112.7155325\
3678212*^-8*I}, {2.616961332192118258743946195435185539657472864328238760005 4\
3715001777566620188945939197265913427964257654875595361388699570054682422513 6‘\
120.12818167963982 + 2.806813544459995634810333084978237502046158742182456489\
0326317150795211760895810859661275870959404074460885338017615421898086785‘112\
.158597937597 21*^-8*I, 0.21746029746714059515038482031701496398488385835572 34\
3398743580866243505220652566163231718398030519993917291538341610567927007 7102\
53195‘119.74227448140867 + 0.207122840644611352077246271189169790504767300337\
7375230439513747333489732847225142376223853726210571058579061823796510987933 1\
0078544204‘119.721122497356 43*I, -0.12983818681813881517274930603490438562428\
8384432272409475034086884927104852223473372690069946190823866236016741909046 7\
42207793257243361‘120.14766044744226 - 2.18634018853018321920930489800802918 3\
7626849717210365336325452623681409704175633489279330319228684939686394622207 0\
1463801427795‘112.37397574337642*^-9*I, 0.18634571769657256211230629769777986\
4681332891940873562601809613762768984512966203874670409609306456453424111948273\
4607706286313176625 4518‘119.68290619201424 + 0.215337315378079777757747075209\
2408910756961370094989653003243196806425253183102812042670437193476171233705\
9671297888998424979033975722‘119.74570606945856*I, 2.514995329285328901867457\
6993906223007739880801790948864049808752899549223127272022413985082515571191 3\
6901752726574710204906361858944984‘120.12968458480155 -
        2.730186522944604984928549760346057771204428209744008115921809778316 1\
9241088468841500672227294216284771533754835655804879441286 21‘112.165339720470\
03*^-8*I}, {-0.607408257964641130356186194563311070235477002709431285720931836\
6430463962215875998369228429173746824550866091654552006358651647405295042‘119\
.59365012260324 - 0.026588762784467855942205170633687438852571706267688658901\
4912176002106247783629478658349405195599625655845871242074073930278991131966 1‘\
118.23486755998866*I, 0.43974682066418154925868512271412182104960619987908958\
6164056640992700412335457087447693159119286888623142423965179055852386459304 0\
46923‘119.84209402920496 + 0.007264466008417459377217137981086155175786901464\
6896414897024757659254325552903061804713413581368915003106779768080629988469 6\
1967432242‘118.0600950169545*I, -2.315026313824762589607682423936923674482029\
8788421838300264259033009249022102538266643500892223974316321158695673636283 7\
818950827285772‘120.14325205639484 - 0.215337336126088837254514929547782551 90\
0488342670495110206677109117562180293307274592354715670725930235727645192340 8\
7010190601767773383 35‘119.11181546081262*I, 0.470653795738230527528169746223 79\
2019980724080158726385721466050530173449969545494091532416811557037824906644\
5878274242774984681178761 8‘119.836765161018 79 + 1.567893328150183497134759125\
0293506514702274476366361802887488936117331767943269448260335441138466432514\
5240715782587103009458 2‘112.3593801075392*^-8*I,
       -0.682642049039426474349876717356507749572267930814196471791553282493\
1943778985436750078379958110519419996561166350490164121518320958063 4‘119.6328\
6934382992 + 9.114454063533251955510362867584282257849145099900908500828451 39\
6774162840367030083246867600095928596153577240671734438597364940 7‘112.7584069\
6806335*^-8*I}, {4.155897137204208546738291899089105809065044855894923102673 2\
6963521805235230156667634050637219794504989866700585299296416953647198042936‘\
119.90869738639329 + 1.115372666679191726175926554519932009936574359970097838\
8618291309574947875644693718228061049594274179760566964737711900693446985‘11\
2.337452594083 32*^-7*I, -0.736702165555573491831717891506997784204788907990 29\
3503171525430774998014932984084971812479231700353258974032089319837782643102 1\
```

```
30096186‘119.66405190377995 + 1.26726445903556712215748458427057674010118034872\
68753090505946067378764071970673305486969641655824348618282890972383740387376\
405697‘112.8996272123719*^-7*I, 2.5149951663347219332546197437130456699738553\
35883516623737970626080834473021231097120846692467341814057512130872371355908\
313462471260725‘120.12968458929438 - 3.434527114449633053850104176073900186851\
10720994341532640804570955924036716825801375409210120403107558680668863353098\
752292032056‘112.26501438392532*^-8*I, -0.68264109018449199657096460643850525\
84970718425761800643765247602286099129527045217829033632458933875637137121685\
43399200203748039238573‘119.63286879837788 - 1.0726300843725304730405491623724\
48518541153732337630532174202440326987831648068274419411943486496131340785488\
77735824177891730455 8‘112.82912634576 36*^-7*I, 3.59873588837610184494581993058\
04870279169704762581162902803507430769878980745225065313413813085126409627727\
18559835465654923441095036563 4‘119.87019782302441 - 4.062787302495827675140 10\
56368467923960864911497572769309562324922774075001711266422883684611439043297\
2900311715664589210791158135 ‘112.92287193459545*^-7*I}}},
    {{{3.598733166956195549799667495773774875231494517594984138262329090261 35\
98465851575259355262410209726786217689266717858911536987774784932733‘119.87019\
788375304 - 2.883282670228948823051987265104541539122443858587106783828353829\
65348780885811659998824717544948651737888964247947245206726334861‘112.7739354\
5895247*^-7*I, -0.6826406573397092052536341858090443723363528704656406 7331404\
81260099452147054295237528258613635436258869195757123656236699926597 16620749‘\
119.63286895198812 - 7.624092039141480162076147769347050061460315112121440816\
10818812674899801523362446210407086292497010598111354356506523360430111664‘112\
.68086493169295*^-8*I, 2.5149926480666992879564897139305241481230817467256407\
63246185936494642555263564899081314893618102977641146504347299345141822947 19\
053066‘120.12968458692984 - 2.42332192977695864640229774801785030632396652946\
44264139327445937648234429412983874123426239064448720186963034986973161537495\
713‘112.11355897954537*^-8*I, -0.736701210123267150001633158941264518134051222\
597124571242311425992323705685213891436455896090259788733370145288766624664 2\
66832704547488‘119.66405178552608 + 8.981144925493989021820614739949701711411\
093788850371328769225296946857611316270531149574360781703126216420796968649 58\
6376501051‘112.750092106787*^-8*I, 4.155892793619350984549475403470079382989\
410820117386662076702400163537685284283847566500436599405438961729786088381726\
637143909976659012‘119.90869737601399 + 7.914490897699591881992346537858059\
138558853887361019577258555551887128823806143662077704003905633006662605968 57\
455248480475111‘112.18845602410379*^-8*I},
      {-0.68264133295941788977513248805666300686685343987014678491280442 4503\
113394947954433918528526649804655420736314710729401644912773717258 72‘119.6328\
6933793262 + 6.474462709036062739201772224595527388540275771889030663054082 24\
060184848996235472037500850929157280330121695702632411260644166 73‘112.6098804\
9108836*^-8*I, 0.4706533789068044062440389268935268715107457480796557121 74393\
448424126183316812061646995636154274186379012187311854095317839267035202507‘1\
19.83676530514455 + 1.139842831844089838724559958003806783847834394405053867 2\
712429517594338924531187609772223385813887849413192470794124649530217 59‘112.\
22090909664644*^-8*I, -2.3150240231497370329291979866136178416914668554722733\
229290631663940092585731624200716248554282750216562164584623518776582582091 78\
040847‘120.14325205647128 - 0.2153371282559015959635922689168592191279054411 8\
313275928489333599248022449710875995849414207413521220745724283638169206070408\
82706447897‘119.11181547138067*I, 0.4397463978845071979629878857212093090014 9\
755294900222858846955399448878193642359801272900399764282335226851439993071 59\
400418516107671 46‘119.84209398619753 + 0.007264457444212934367725441065392661\
626933680255322150512524177629555923834983321347037751349483574183102077128 92\
414059834641574972279 9‘118.06009487948741*I, -0.607407909139537285559761653550\
979684270223767439602547827053859622407059362166108951353278341606459817797 89\
719917938168061649851125324 51‘119.5936502926703 - 0.026588701696367132597659 6\
482179344082603506983359405670683602641477155087728861999053708361836644771 12\
312166192105144060283205248317908‘118.234866981664 81*I},
      {2.514992770602590699642168819095970785756413592414296631980339197445 53\
128955743886049303737680051413902110174798022998674793146558842339 3‘120.12968\
458381316 - 1.975643400197296139706218218506787502059868343533992618997951417\
078300757915696829635960173633677814708728657458761709411943395‘112.024856400\
82115*^-8*I, 0.18634552980183691420979851754221659597902055187104986471918095\
302168699713529531712632917156404688399311287427620865383088347000869051‘119\
.68290617827543 + 0.2153371128561363932927447455457216075135313655104256504 42\
904421608389922193816703680750077732838014125241707521894500815088321953493644\
4‘119.74570608517607*I, -0.1298380446658653260394718288547594544284429632617 9\
129899049560618028638885664058388447179658212727146493979606163985930356545550\
57270409‘120.14766050687132 - 9.812371667579138075432496475724705683139792126\
374735975016113586191919150610352717741443087588189270316922484731082907166 2\
725‘112.02603253020082*^-10*I, 0.21746007586342770591524471480251913239240587\
793064707894631927126375442860990648939276533977123466487464110458678825551787\
828313105911697‘119.74227448839693 + 0.207122637705778828306092459000841560167\
751046889499303172683028325478822899844674474606576228094919610846079595162 35\
505778204786415461‘119.72112252139263*I, 2.616958687575873496369178994304859\
718188572282925409218215463593540551192372584188556951803734500361817030 3102 63\
368218621248337395228038 4‘120.12818167701695 + 1.9537418426076987078168499662\
796227839978637427882586241821266627575750912416715669078628270108706904222 20\
322829172717283947143 7‘112.00125198801652*^-8*I},
      {-0.736700672588127249503517994027228620456188864527363544694014 9696089\
212498774084456972969568775960705301694484362156151622521941848666‘119.66405\
149753737 - 5.879668795619053941159785879352990770011609904267763978228788933\
101340806994457817812244737091559924655866204677259789533458 3841‘112.56611329\
394514*^-8*I, 0.4834772415987782806536757902515194103636252272114496802811720\
685488732421782557430021016444744981846374872285933582735396355183331701837‘11\
9.88100855730318 - 0.007264452865440492869068674035487447694469340197199553 53\
092561137568510215492541360791714477314254609312302093675460122719906555295038\
65‘118.05783543189905*I, -2.333943976353062183334622357954822716304755443221\
432104631496563450567711509845702397875789826656840388399449148664005020254 25\
074916823‘120.1440286221726 - 0.207122620237098915136608935602700825102865460\
```

```
0903547501268368390573985167731065264741810088540748738119023341518933984606 2\
77378223360 72‘119.09216572665063*I, 0.46241170356188591652621371748164378520 7\
34589056509028696499652402562633690234623668880973344193530753159567345243793\
45004038710271747795‘119.82055960957902 - 1.728326020708811274491056374803909 6\
07731621848126603796086437549646378955607283250629093257002932716916946159144\
37748646545272 91‘112.39315654159762*^-8*I, -0.83087501843168983883000352200 23\
02259939523208933305434659214915687577057525593475385209181489112991366590781\
11771232318195721750033157 6‘119.7163685624 24 + 0.026588696575716916227104192 0\
06304413815801425783339533745280833469758013250524266238847482617540123191328\
19964201162294118604380985075 6‘118.221529908988*I},
      {4.155894402748174765842987807781832473398246926763583561172154861941 02\
49325964980745320064645082875053444944500178480648262520219144706‘119.9086974\
667658 + 1.0828945191398611117849763344077007432696639601222890496415704696461\
10769993666785425934193896784053370346667456136929546952958237‘112.3246191183\
6474*^-7*I, -0.83087498694916680059890675480608912459043550110290096941959730\
64318221637811535781988400161685528208172750783114560682713233348119423 16‘119\
.71636857352799 + 0.026588637246521230586959256740795790577603081344736051218\
52865966526359401544676380075591133759035861398914578894007602343293820769860\
2‘118.22152896747544*I, 2.61695893349669109526747625156034511828895649045272 0\
96413760800277507957908796859303455629362723377303358947290953764976612749872\
7581445‘120.12818167570661 + 2.43152996537870538045747128098976274909539782 71\
43317566414273437341275920425951832256617119748670807174518665456260681254518 0\
2661‘112.096264394198 78*^-8*I, -0.607408102635911356101990415500068875804803 79\
07003869751524046066706061264314955165196239831829604495516309517060376639224\
8690420505871 11‘119.59365041999507 - 0.026588654363289094651317901554413621 90\
41493127420624727139800692718639738470994892766791374582134965890389430821542\
84804695725574024726‘118.23486619751075*I, 4.155891533038719504533364534161 51\
11076514723377998541913333315248025588315453107192531899481292481419351367575\
89899719514435234285974 99‘119.9086973060922 + 1.0828942737549029782931754525 3\
44318213203131516128387351710557103029062043915640291284824766937826631500629\
78939114454215292455 85‘112.32461915916672*^-7*I}},
     {{-0.6826414033120674039278274362870794750657411697894326953841999400117\
05495209592558572051753737353412620294460708545744316955762489416000 1‘119.6328\
6936944047 + 7.055375207916364517270318664212468504300103338391930532563098887\
52357129860915618877608372756490511449162240156291090961714886621‘112.64719685\
87055*^-8*I, 0.4706534404520158566991311058073287971893889387246084808551585\
44861282826064374250203916079865100620441723220180427780900535208476205 65‘119\
.83676555644492 + 8.90165663303897289645233046273173557673467050160160384020 2\
12173130370111635184620894197615736919406481349885276314200638156146 7‘112.113\
53515680396*^-9*I, 0.18634548116596111092158159875128706583650339649998771114\
41459888070184682754764115018608053287807891095988514833218500443804580192153\
22‘119.68290610225135 + 0.2153371277158304786938444195469265252783653280515455\
50300756250557995400356067533677810152238595594533396294205272796878664866407\
3577889‘119.74570615247134*I, 0.48347728735405661120046968251463177252452826 6\
82186207399225525827330188326524569394776416821427927956933665267635371769490\
701167009009‘119.88100841181648 - 0.007264475541195025946317063844933070465 92\
43694949100291488259517196272333752144777627658387682979212769661465397147360\
35959542672180715‘118.05783660094569*I, -0.830874647101472530157304493427987\
66529484337828001050354686373744986212662426660939461443831227319180551699347\
23777892983794958080030 7‘119.71636841544728 + 0.026588595722570835995287607 18\
08716906514354571958304234778898869565630921795019205948463721700610812486464\
42864986726049183032804732645‘118.22152830878554*I},
     {0.4706535336506524802093171679510722142050018116835477914836375831799 5\
28852602147787532916249967873132005449368527025290922931207105848 71‘119.83676\
554087384 - 2.9046614253988991064223595098153142710465128707903822073451803 00\
88561369649746774512156163217280242402309938151858811582976338 75‘112.62715973\
439477*^-8*I, 0.0520280533114745157122014124912679915962476267646437663335029\
18909497529713390570260913212494527813152850838168225335358595878130994509‘11\
9.34360520350057 - 7.81595475452663298171125190774460323743870448999769534911\
67973652269651900877654863081495616685190620404888057413172935395659‘112.52\
034966356806*^-9*I, 0.16635432120957822737236765950094713129582475524233679 94\
73264630351814189232853744887123779274859018540416924308197368008661325304608\
921‘119.9314374536839 + 4.36852264050523451110574559194478234299094546655136 1\
597237539933300539597254538444778974356112707418830205380368757237262113801‘1\
12.350737957973 95*^-9*I, 0.10561435095567204816802899674983389273234500477943\
93592895404022724131714278934583660279259299513811726940758283900215153092 10\
81806841‘119.6691168941298 6 + 2.12356452855262887942702293427390473129061958 9\
85785315674289942695666391666236956548559361715545719276561358626504106514754\
9509‘111.97245942206102*^-9*I, 0.46241164451331349818962251224371428045436961\
91159946098109736143065067746306839395508621460507423362572208107009179548229\
78218832880369‘119.82055971756414 + 2.35874791329722868851158077453688354264 13\
22068732117967206030565502430882544837210626416880300022545279414809673155006\
61982505728‘112.52821248451872*^-8*I},
     {-2.31502413021041458132841579672712031857206447568832722446629400034466\
20842246995672836979577145048421598578697080693747838471853264602 32‘120.14325\
205819516 - 0.21533711490015228070030766295802498074157200899255026966613369 8\
48394068928960680003635033655002042448903919339483170825300678641706962 3‘119.\
11181542608408*I, 0.16635427833612131574737808401026647172979365843859603618 4\
00545982776156005256138151127889004592642866443730160728558365067650138443619\
5‘119.93143743623116 + 4.14012488727678303849485657469288410117096319181407 3\
76303036691677483456492585562533671244382774835854871609677895840203867 54‘111\
.32741690345567*^-10*I, -0.4447115939453458681315655084836976266089430124596 0\
17617140102051465220902455052058539144381003555286777513790338693634347603255\
8352474‘120.1505095025504 - 2.22149768032767803195098004653260035174306048568\
46252682067976921872763180079983283403530314215446743848763464598686577162408\
1‘110.849076914679 87*^-10*I, 0.16848002084065999123081856227946001145758536 07\
01174346086955790089361249806316000003078234796966901389456681362994881766536\
102077201148‘119.95871003064028 - 1.12942658484479063832340316635075045935298\
44920740995427357339224364178160782902888250911482001682470354040097181074534\
```

```
0831406‘110.785019629176664*^-10*I, -2.33394392946616720804783274913088892983O\
2885214823101229545997566301279627033448146865570187386186790286809946799594\
6688129951364973471‘120.14402862134754 - 0.207122634374707946204111469713649 2\
3115048898928615324515262751294859762921265696393800820508161045360050646570 7\
6182432675364467769446‘119.0921657641939*I},
      {0.43974637545026411533834317460797313582834282852493022054903137788995\
0058664031612825065291054625577556610815674418094891672943965570841‘119.842093\
90012279 + 0.0072644681608981890903489570454169062468165715918140298949323123\
4116626789313204998732824190484311550366692174559456632575271737188042 8‘118.0\
6009545624897*I, 0.10561438370368493802163891388142115408055177181110623060439\
5529285629167777561973547124518657947422334688863456395378894232724664669476 2‘\
119.66911702777567 + 2.37026899159915333698148391397602925112355271759839870 3\
0217112635218756950958484908755682291704389361653749969480078947380123751 12.\
0201915936658*^-9*I, 0.16847997414428291559240847748715113496005428949918748 6\
9562501107643410467777563958060030756721259496636569023110598723213374430682 5\
2049‘119.95871001933277 - 3.631476321677396684822037302785622302801319989096 8\
5115385022217911651367906132222144317169669235612240443136505120731340415462 2‘\
112.29224494891564*^-9*I, 0.10561432115736431656793552393944256769478316611906\
7270630950762971008125142522996708859714318556565471961499062164093880298712 6\
052663236‘119.6691167951111 + 2.370268352378563177630395718765864956283544537\
7260181781733996341590063345768206768572068159984766255519918838520500569798 5\
7144‘112.020191510074866*^-9*I, 0.48347728133443570185074615513841661547719273 \
0822321920628195895070436822494947926866055034007497581643688796150998391626 4\
84453675003369‘119.88100849505713 - 0.00726446232773760783495725664543329148 6\
6994296077970667226541820104138824587255051163502330213348313647434262015827 0\
0923213615411951061‘118.0578358996485*I},
      {-0.607408432493629488684623883739226964814013398335958666831362505781746\
6952391885551249293363509881026100426525005877655016413241074300018 5‘119.5936\
5068507867 - 0.026588600388553058251616937764685577597298396568466952785617 46\
3577947151429021333741023399719094145683095334188251570328206561421252472‘118\
.23486534513252*I, 0.46241174513372476039994062637170067010262891606716646125\
5675395236946704363422560475356195250243731707444780512576875340187224620897\
08‘119.82055990885328 - 1.18010405474058295521165645240921137917745085602036 8\
0167510718440925418866836224748163390809657042687130205891156658619589976412‘\
112.227451354904424*^-8*I, 0.21746014233462113891720643613475620862640809911 90\
2660317402214268817717070295166094715444805392676684959754254162050914363777036\
949269111‘119.74227465398712 + 0.207122618474576923707663091785134390311417 22\
2363292216212850943740382024972120583236118527470999620925492355426694626624 1\
9459664214508 4‘119.72112251390776*I, 0.43974628854900594308799327765427383894\
2709188055488823697119267674284552678261658506683797508693575110022168711924 32\
8589700229524150263 7‘119.84209368076272 + 0.00726448310541403729022969881439\
7022431364303558146793793089603330547496270680711264158283800744918734980585 6\
7768379080860179306364 38‘118.06009621614555*I,
       -0.736700388711292818052210756460994511666507478365374070626584899551\
6673872389959805431213709037594043105950109143087264876509180092436 9‘119.6640\
51308275 - 6.415114271426936138111513494251117020413992926314238336866919158 4\
6595238210906377229261517085540403539133701167474243675636362 72‘112.603964806\
0693*^-8*I}}, {{2.5149924634620148483996823694906837770970921958203599809316 4\
2624432640034022619786895385780202025977318830452472500763775972326445211799‘\
120.12968458393695 - 2.164218966164524751429883756353171478683547582000610297\
6774976414459686248653128751655528981900868178998515399948856226100954189‘112\
.064449094634878*^-8*I, -2.31502400254542682740700896612970349718331504060905 9\
1332120666899609546375777506647874079894000409625731504493916148614943893139 6\
6833454‘120.1432520558098 - 0.21533714389302343455010375613887727520876189018\
4741499405249609204090254268675109923181103971717420935543911356049197665948 2\
28604745802‘119.11181550612164*I, -0.12983803921825050897901311249916638132 14\
5438510887875765715349053018430415684775350753683307563465585792408672680066\
2754022074889827 8‘120.14766060672143 + 1.934842181481414018142885165067732795\
2890110739330867100461859081867943850250801472797378096955028792374694614392 8\
8053311201 88‘112.3209042053316*^-9*I, -2.333944028826902992118174495655492670\
0014957267918665186761320991538666965262953784185109564541297873814189594495 2\
8373898158403696223073‘120.1440286230966 - 0.207122606981418091155989661475 10\
2180210772671867559826142788677512177397273686266519375834004515450697668436 3\
8513493400082410650925428‘119.09216569001593*I, 2.6169590066652286675826090\
5702453040421796353870206287866758772271406308056924881174223521153824033133 6\
2516777535370602445192390346034 4‘120.12818167804149 + 1.765166815119600328365\
5327602395769791951380516877882436227640222443307444557521473431173133633030 2\
1646175722794668063948321137 5‘111.95717051252949*^-8*I},
      {0.18634543003042132591012916793937296706061216847280828474571480526416\
4752402065468700172518420958856172890358511756209823072432499285378‘119.68290\
596434494 + 0.21533714041288537222222995761117974808391529698051677662109456 63\
6710421105444448296693549323633429409431289080963457118922449630062779 86‘119.\
7457061593483 7*I, 0.16635431710369411630656845185865329503307373485595080408 0\
1689058748987538946125365010044664052843034406896766351638289771683708502668 7\
4‘119.93143738438394 + 4.7407367592098846435033676906480729430821881481375006\
0320309174441872778375078892870482596873051689456751264737751916720821929 11\
.386249149289 67*^-10*I, -0.4447115851808933072780344299169914240195965790194 1\
9584922310893226865488428669612825190824593539894766270112548462936663769214\
42060448‘120.1505095025743 - 5.3156221964824563350492703345736839318517884350\
6870626569886841587941995132728704384587049946505936388845805665935955773988 6\
8‘111.22798516541408*^-10*I, 0.16847996985029791190244674601968883866319687 80\
7159761941245759552865082216563408023223964413008668051046747706297589693867 2\
6596595114‘119.95871005567344 - 5.288185204017393312261164215505263605134 01\
6019942579836940516332752371487254601107080539339751422142304825020487488754 0\
670701‘110.45546843625299*^-11*I, 0.21746020434810390844845703896673423197779 \
4879267104659974711571144810410439309949493983360390632916992275073718330886 9\
1854191792148039 6‘119.74227457582811 + 0.207122608876367296441065770363727 17\
2851530955176172271078166921276990679885947372490287657677324155796503558072 2\
0177235650873056479703‘119.72112247377473*I},
```

```
      {-0.12983805040547001765642648685261722788868954792864124888853844846674\
47096450112766350042559188659747453270006401034180300652618385551633'120.1476\
6058893777 + 1.20727927323755597039415289508900302637551455110891441152872931\
83276952014957032699063517027511160550375861874786496714471167'111.1160663479\
1676*^-10*I, -0.4447115898155768614613736198084119628311406774226274720913356\
23652035711875034294929851248180822203323578518576694680575530683976007813'12\
0.15050950257393 - 1.71705665057638413589986946213898310464887248253512236283\
50484758806882418293760553648638879069264722267346181196030120500608'110.7372\
1567865248*^-10*I, -0.30521826190614302154821094770563096575379373193048214721\
72438615059079813992336668859352323302938989583906383788942678635152785361067\
24'120.15051499783198 + 4.45036021781852015609377716706552264689388981338905 5\
26084486647517073887681876463104059261436518565748918518836154151840274252'11\
1.31429964769198*^-10*I, -0.44471158981557684815924659241087454980154717114095\
54525412239695779323157979008130639224339342725901878828657708256638065736892\
24315803'120.15050950257393 - 1.71705665286224149350254520937724799283906029 6\
02920777039629654761772530009086881204857810707562172235035267952685588319844\
931'110.73721567923064*^-10*I, -0.12983805040547007347220364891360740628233 97\
46923343538217002068414968231355498400230556854247876650405285581262044590839\
029415947287621'120.14766058893775 + 1.20727925347869078567872429012999312196\
25751161301231255505086448154240601304154411582754170195012237543920608233987\
0349127482'111.1160663408089*^-10*I}, {0.21746020434810398505374400018003399 79\
82538032930764952675269615084029830368987957774016279457748125977702148914413\
23452162981601124604 75'119.7422747578281 + 0.20712260887636725111937538513558\
33694329826999554508053481744996831535115790948255352312005071336697203011533\
89608444979832320503014867'119.72112247377471*I, 0.16847996985029785699236640\
11606789473827916802765762763236991775501904428046165721738209952401545757914\
51248516890676951162983751219692'119.95871005567344 - 5.28818445359548931107 16\
45580776988550764344048576663841802716139349047022403476370360910696877635267\
21081841675617755659823492'110.45546837465378*^-11*I,
      -0.44471158518089331668919564626023243907959261795128200783036364665 67\
62240074126733827922795422366712342342332883625305859133104171965713'120.1505\
0950257431 - 5.31562252023712358161504355307576845514414543450967616084116267\
11894926753696244224930912315484955947014761994648555104624259'111.2279851918\
6536*^-10*I, 0.16635431710369413482264748692203443501377707986728117939802760\
42681919353452744784250084393103514959278368208299857623762764233014189 99'119\
.93143738438395 + 4.74073663953186168613516928447562645008156215870038178512 0\
17988284648271416327845368688333407082770562553445060087136510334253'111.3862\
4913832611*^-10*I, 0.18634543003042137737945443268772615053625423825231377488\
54584707149635303498824062057754749914504070456910653879579022899155403350004\
25'119.68290596434494 + 0.21533714041288535560411878090359742067973448888581 77\
55242510961489857875000450425929704485966226293993555066254472788792385260223\
9130655'119.74570615934837*I}, {2.61695900666562312544374602514983519245613 58\
59139120576349363852301954300389724191986423465955067015557727868025748293 29\
3168321589435 79'120.12818167804149 + 1.76516682017873823453518755693933395344\
25456568677524163291121857463626248069131123862668206830140812540444444397999\
537458142977'111.957170513774 22*^-8*I, -2.33394402882690300078344913675069571\
15012205087397140679450890339575014698241273917012670749111139389612503402028\
11626188258521466468391'120.1440286230966 - 0.20712260698141807044836487617 32\
14483380508676720851498227523805247926060092199856074106302458421499460273015\
52170865770642424910526463 2'119.092165690015 93*I,
       -0.12983803921825049986828905238349040095291427423909517897294237033 0\
91693427413937704046108738043035474956905228638127778007960739441333'120.1476\
6060672 14 + 1.93484214679760204091032399723466515944806314116326425803393214 8\
01189550272559399554298097371502279528176311409065884425012356 8'112.320904197\
54648*^-9*I, -2.31502400254542693140661732549015326056741625535248450410250 47\
05420629468814495687244750583719739630319743576080655662791095137534739881'12\
0.1432520558098 - 0.21533714389302342307713380171998519815414961525446780050 5\
64577341306274145096518203456566762320092941677940604204440741329826107012725\
2'119.11181550612164*I, 2.51499246346201450918978246536841397532145076754432 5\
79549402078772026176216005597375636729379704724186271767285481247972522789 00\
8814107'120.12968458393696 - 2.16421896556414382371249047330931425638401817 55\
03900158497641860865473868594420882003580366212753365058111097420391173032241\
6281'112.0644490945134*^-8*I}},
     {{-0.73670038871112937399141883776541166461543782230904257581940087986 15\
14389595691176021050255420603413158875206418451245372165657588782519'119.6640\
5130827502 - 6.41511424767165567007601900452017040709423940310310479880522300\
40774702135013223726864440139827218348867791963061440637883517051'112.6039648\
0446111*^-8*I, 0.43974628854900585257137533884652502707840527813585275125220 5\
14038866674085270677655453410077693749729585497423917830259573982682465822'1\
19.8420936807627 + 0.00726448310541417660719604159670488033497780871643268171\
91941862515197400942581545395241234597225794554350369811399013474784257211804\
19'118.06009621614555*I, 0.21746014233462139887613440492902370689347953808047\
62539262876591188933720605868873264244988374421767705633740934908103745898782\
40468412'119.74227465398714 + 0.20712261847457691697111120769560132295166589\
95610167363176388280027686886233737111832537711448661320250828651010125753769666\
9132265029846'119.72112251390777*I, 0.46241174513372498053563065107178891518 3\
06529834650221921691659097409660438202377995903107381060484369511799252868639\
11336968157740508 6'119.82055990885331 - 1.18010404200849025172713179671774786\
35131425084937551515489492153383664215509767239776996838752483389363565855587\
96403225060193 8'112.22745135021887*^-8*I, -0.60740843249362943142266532444114\
84579515718779287336864388947321746812777891562679948072633994370375726640013\
03821525636295520945310715'119.59365068507867 - 0.02658860038855274601768080 9\
77907666147703711701617204007341561189662940472368515319198165144895903297330\
41234345584626571597345912976 44'118.23486534513252*I},
      {0.48347728133443559797553555755123506116483422131065023810928238541351\
29290346481074698125256869190009064738722600807425187392048067409 7'119.881008\
49505713 - 0.00726446232773754729576968754974220453472848934862534779848387 72\
01268209658208854975281097142516252259056043312834921166158278064796377'118.0\
578358996485*I, 0.10561432115736439349627392013472678410259726627664494940149\
```

```
4461826930217780092242174956519530051932147534911385472911755024610540881918'\
119.6691167951111 + 2.37026842422576642064321339916854532246605193802613281 27\
72865073576729269190944571971745140590781653956403665483710561216601887'112.0\
2019151423913*^-9*I, 0.16847997414428292116094424816344575396208884206758072 5\
45711253139152824642809969792511297176543995713175563916150401540551350767869\
6601'119.95871001933277 - 3.631476309229412800947665408696513775396088396 1412\
013529669059693475292878073946750121475395360515526459789359425109443960182 58'\
112.2922449474269 4*^-9*I, 0.10561438370368498860063294606186769940668652564 59\
369085083855054949223765197698016980923499771979417468905546018646479181861 75\
446956392'119.66911702777567 + 2.37026904022692026176883291071708993011189 07\
094071625431147548493377217726064519439714261775354488429970543485182769807 55\
93053'112.020191602575664*^-9*I, 0.43974637545026394912881990994376919437444 70\
914155919310557078750530499363673496159805597652616726993948838413398919 3546\
701586880490972'119.84209390012279 + 0.00726446816089827095116870031177862 343\
888595540758919961638879263559871141229688922342398105248989879688578599 43904\
09972602078877447691'118.06009545624899*I},
      {-2.33394392946616711382700370250444490068751158636078876330569264870 99\
325749293886115387473170454954717551404986735417593455000103888187661'120.1440\
2862134754 - 0.20712263437470799300890234765362313842755538650285634064107 785\
648587423423820969030258482081748956511366792612079263294508596446330286 3'119\
.0921657641939*I, 0.16848002084066000642162574246661830793003948562423732 3969\
341646164294135743348142991067616175311861452875234248986150311240641457 1232\
4'119.9587100306403 - 1.12942607464142280502703130554615890080599992244144 332\
517341728603211142694258139681106350732482611683434682198809305603085 02'110.7\
8501943298991*^-10*I, -0.44471159394534583264451255824548958593434756720 09781\
401041680935445622780747748059246016065781137625475299111667379217017403 99136\
635616'120.1505095025504 - 2.22149739906406929431840768081856035885257 12828461\
585337425439494103314765205290178426960174808134319306316675294464445501 31453'\
110.849076859693884*^-10*I, 0.16635427833612137047725000012478322325824 3090951\
578380699856048571254288782015561751555406697768687459758459867533891 06976006\
7796630811'119.93143743623116 + 4.14012531592927906601758859380027105965 16710\
769838558293840480405268713382129290085959460529209833111418604928027690 39063\
54287'111.32741694842083*^-10*I, -2.31502413021041439835754941643731652066903\
115608731640595124733090420097380833331354540257097295555205552408275411 26581\
91111739583673486'120.14325205819516 - 0.21533711490015226942991618534461663 0\
580581496409490010235200693040779831725331046909531348084378427447800370715 09\
565506631685617151967 6'119.11181542608408*I},
      {0.46241164451331372102653180343070312020634705168150259975674174548 502\
301963417942914921735076431583319647791487706655473015684249067117 4'119.82055\
971756412 + 2.35874791815333905599979303378665857428618829984899914892613 699\
439156370995902118671300132255322085680858040501941366198286658'112.52821248\
541281*^-8*I, 0.10561435095567213681266702446403930037218449315497851384 68318\
508291507003442161541018511183403942534233955591629552762617208142998666 69'11\
9.66911689412986 + 2.12356454135190438950784289546421561426237820397527158 931\
628466365133464383266430394627003343337067469753008910099571687335950 9'111.97\
245942467859*^-9*I, 0.16635432120957823374755513001946214148815489696217711 69\
638992265206563248331584812039846972574054500156338099606989918347585855 65434\
912'119.9314374536839 + 4.36852265201292163220347373176791567350537646685 0394\
934362646169453472767038185896821294102160398153118582304148539234606882 384'1\
12.350737959111796*^-9*I, 0.05202805331147456681058627510478116950921996535 930\
012753748057679356896610806203503098914541652388878686442319344263074311 75261\
36466813'119.34360520350057 - 7.81595475667121620840205279460556878522373 3969\
328805847854765934936449723378314448148130180231592187057972941763630188 20964\
5976'112.52034966368723*^-9*I, 0.47065353365065267038010680117590889151 37083 4\
579021517208228189693638073142758146920049640819756610116924339109856701 9965\
07092626261419'119.83676554087383 - 2.90466141896823102927888008959033652 6990\
072971808609287025217334380593081197717592131432794926983574171707423659 08794\
70097695573'112.627159733433277*^-8*I},
      {-0.83087464710147282035511402925530685319633711211659206875479953919 22\
470055549059220914970542510801549043475317555168875497102668351635 535'119.7163\
6841544723 + 0.02658859572257091451795640805477154170542503043558424627372 917\
150400867061411081424596872374724601801761147610379709676433141440308293 83'118\
.221528308785524I, 0.48347728735405651062291441127543086801224926516666 625019\
157469690799387780633700012989143193713323803681397733645833780129004431 62579\
76'119.881008411816 48 - 0.00726447554119496057366001298941265831116496041 39432\
416684866495763614869253711675186083218948489722506007819314824113306554 18813\
2382272'118.05783660094569*I, 0.18634548116596130747273052783163232174916 4096\
804015190712880430464382615693604035718152929155483853566071409264756416 24906\
152834978067'119.682906102251 35 + 0.21533712771583050646547452008450306417 732\
094335628378972931628133739300525894207182193693389782924205478317037 8735594 59\
074340506335777932'119.745706152471 34*I, 0.47065344045201606897706839682241067\
527741095713096089159053144751099620088467056667394404018620434583470501 05941\
896285435850561357289 32'119.836765556644487 + 8.90165671390250228922075127 371\
158383131011285084062285612282482676339782496405869988804933157184979717847166\
9519226695017737228'112.113535161602 29*^-9*I,
       -0.68264140331206751299571891590610556710246830983551623836896048913 24\
658595743001546067829586294305253265305714755317369022833636704134 13'119.6328\
6936944047 + 7.05537522112973900897090888357804061826135725611238341373777968\
436672670305071558243254190170449190095614153548492410608456216 07'112.6471968\
5951884*^-8*I}}, {{4.155891533038719183684984786529787279688023168560294329315\
120511968193652201076178806602323403317130115499949993784868448252888026215610\
49'119.9086973060922 + 1.08289427581728303767600008110551940698862316664 3546\
962658502275602486472920401885191677444968309734943096922183833360786519991'1\
12.324619159993844*^-7*I, -0.607408102635913013538364227079827874176622926 5455\
183783698032249476359069692537739087153395546079459247525662667804994868958337\
195480036'119.59365041999507 - 0.02658865436328925925125122863288770187968465\
125136429304007950467041148563948205054398069304555929641812543084859469 61700\
307582933946871'118.234866197510 75*I, 2.61695893349669092666691193081813490467\
717887607225394074372929501871109728112444547522847907297124692189402973396 10\
```

```
4462389641903319609 8‘120.12818167570663 + 2.43152996905851016398585172913642 3\
2949200327801392298346073095816455745116281918336910508887907754912844462732 8\
5722811816562506 4‘112.09626439485608*^-8*I, -0.8308749869491668297795559799413 \
6847483643376998020109768233741405040333546702412102289102948737319692876201 \
3638086901912529428317656222‘119.71636857352799 + 0.026588637246521066280883 6\
5835980345940567968440967406423846476119448102893698248201241299277976910449 \
0112187017485326313432833748808078‘118.2215289674754*I,
       4.155894402748175254365267433201401544495957075449112974389359477866 55\
619512904048958357453991931656372791836369779535476112237679707684 4‘119.90869 \
74667658 + 1.0828945197895469530851942103687538654870955158481097882333337385 \
429741297895441450732440227732886123858639135846919114319229642 9‘112.32461911 \
86253*^-7*I}, {-0.8308750184316902017761757190445357202439464741013757711416 1\
476694316544547492752738722777952478175679719086526785510310794932248350253 3‘\
119.71636856242401 + 0.026588696575716841053415426079250205044619833249761282 \
519010777115922401665941284477237379954245932121846633968874627283561080373 70\
2017‘118.22152990898802*I, 0.46241170356188613786723237172168637218860636786 2\
897246177534073276614087809553602572052684567260271461456731805787626566724130 \
5936637957‘119.82055969057902 - 1.7283260216484495893596702617487144372041523 \
389028814786762728404041958883685380121501660094056153638097600418668570179 41\
6935206‘112.39315654183375*^-8*I, -2.33394397635306237830625649412861121961 99\
744964190632921583544055560255329947983724731646199871661684432836297176090 0\
690041529290729735‘120.144028622117257 - 0.20712262023709896763086415701654883 \
369265757505394027399962454246937613721679600957618611613452647000119008190 02\
624093766883674097238973‘119.09216572665062*I, 0.48347724159877836905712888102 \
087707964129534342768214731173701269575352008926157056765495894336953180642 68\
777801836397591348587092899 16‘119.88100855730315 - 0.007264452865440592872124 \
130154112902122734242897284086861842747712714241134115218158046470935079559 96\
768636332624107546910169288914493 9‘118.057835431899024*I,
       -0.736700672588127350340283915186928386995342395439435085784284599204 3\
266551345315662760473839982970289254520046449488978100638174099712674‘119.6640\
5149753737 - 5.879668808363305453246834783714572012086961316930466055365010 00\
279204679768168045372408350880726879354593059371831067650659791‘112.5661132 \
9488646*^-8*I}, {2.616958687575873950889615513827947770317424196182611745857 1\
900154384531628919239711673602073616762508459752158486766598836793237752585 09‘\
120.128181677016 93 + 1.9537418376594625095002730399721963782868530658538782 8\
371902212688999791317209756746155031348001094624916902686576014339673344‘112\
.0012519869165 7*^-8*I, 0.217460075863427744408078633245011878212488850264062 9\
697555678793924378296202225592229519973698393026431475770554246314492903638 19\
281994‘119.74227448839693 + 0.207122637705778806054373609263678969093329170 65\
800517428377346403154420458348033008524374695236666159150909348288366863494 36\
67066375829‘119.72112252139263*I, -0.12983804466586533684291308790432045037 24\
905835292113825437316365135605734769447730148263555638920927649820552726328 34\
098231953620063647‘120.14766050687132 - 9.81237203826916750851250921596267828 \
219912907914339645499752852430250128248691023667357620085092219669787847993 232\
50911964210 33‘112.026032546607 54*^-10*I, 0.18634552980183689435476495224300 94\
111207892667308780530611157232697746335172542785695344927904450270786759095 32\
845416895331701236639553‘119.68290617827542 + 0.215337112856136275642614364 83\
135869135548033365339338999039550924462619970539645906906470363105194329395 9\
722470978456783988859920243 03‘119.74570608517605*I, 2.514992770602591213047 64\
216750240616685829235350953222026979057860106933606997752582020570798399794352\
887713000820880904086561033494374777‘120.12968458381316 -
       1.975643411546018565826831354237269647859413762195592200715785923395 6\
262695410271624042046443332511739386096410235014925654343292‘112.0248564033159\
*^-8*I}, {-0.607407909139537104883196047580778171915116282689220855212082525 4\
917105312726900597955159045326620656581515490918982767773887901651014327‘119.5\
9365029267032 - 0.026588701696367014803194797334848907565026350994322170252 63\
480855533519889100600018514876941645066941514387945818339256134481902135 384‘\
118.23486698166484*I, 0.43974639788450714287012588210594610668635717477125723 \
828264687824432387515213272573534905378365511446983883090681122930224949367 45\
69731‘119.84203938619753 + 0.007264457444212963731975441381272391395537520635 \
850345750718601481926073964179721701285131435333733795095023801676362616280 38\
8664540487‘118.06009487948741*I, -2.31502402314973704640748036591193051342381 \
882207703252095680058279577836547497158864810857056140825751706561437505557 71\
19593093748507144‘120.14325205647128 - 0.215337128255901616343709862432424279 \
303025144903721523407400865812264509915604485115470509570481907420038783080 65\
332387070551941640080 8‘119.111815471380 67*I, 0.47065337890680422993234185 0882\
355951937650153389487410906913220674033397253906100767634796254342786029486 49\
575731649140792846248542327 1‘119.83676530514458 + 1.139842833117370606654520 9\
617268265892034502620754259464658320727291221631933861498174043829037813384 \
495567361948000734842169 1‘112.2209090971316*^-8*I,
       -0.682641332959417589407705791482259697010123863483856133148824247323 2\
051333309256804069460238787827167339829846468074245409455797186149 65‘119.6328\
693379326 + 6.474462711793319812250717869678859401176440584593621014879048978 \
409428990067075522698659463446701989537791241157533019554970944 91‘112.60988049\
12733*^-8*I}, {4.155892793619351070416956085216139985328007685431076280056237 \
621838821632774894606611107713018529699094028444760356015999244737639090728‘1\
19.908697376014 + 7.914490920898847683028410075036153244221114287727330895021 \
802248531013990426695202966525185696604921107779853618133473100011307 8‘112.18\
845602537682*^-8*I, -0.736701210123267506462639762809867792743104713463667200\
789086584550824854481807753881685518359384759001375707499648211443180639114421\
4584‘119.66405178552607 + 8.981144905894312550656797903604675710828286584579 5\
741527656407521740710482716840811495502001271190753508762114060274193201184 75\
9‘112.7500921058392 2*^-8*I, 2.514992648066700032313833955090591408528231008 \
635098024058079839878957679500112487027102390899397470164777826046661991248642\
57684769727‘120.12968458692983 - 2.423321942742267984577842996278113278531956\
233957707719004885293796783424196068467841243062187507599615508249337467325 94\
0088446‘112.11355898186895*^-8*I, -0.6826406573397099755231646854727565852025\
938947851633994483930464666190327722577195584993449905225340571772947026482 11\
299024915858017751‘119.63286895198813 - 7.624092051835142996272267493585181 72\
```

```
0754264559139665428935041798341638961528690856806143717711191879316742101693\
1787518150353811‘112.68086493241604*^-8*I, 3.598733166956196148785791532804176\
8469871088777730834339926197177137214845356092625447675470817429337155428161\
8403338674669525923375718‘119.870197888375302 - 2.883282670235822418160900309\
0360921561211424950588448733871214252035634116237876888844290803487966607839\
8151815857972628020862‘112.77393545895349*^-7*I}}},
    {{{3.598731026324083378492404777085418967158236396970798933693955258753595\
1646300032120916388521477367948758047563249007387312295838166775902‘119.87019\
798485744 - 7.341425826672244025290452208828039638174003366431889118087226061\
8577533635738519569667353423559308075802429251757582821820532469‘112.17982901\
200672*^-8*I, -0.68264042045984174138147583086828030094552831345037760763189\
8640246300924397449924105558524798141265846921450735517989620374681122576851‘1\
19.63286922678103 - 1.942938507270512265189412457359893348048778172228478208\
1135648498010133572838972307619881795117557616333609411500898618905321971‘112.\
08713628234241*^-8*I, 2.514990170707305382601127153953688048863383770184436236\
7863078376276651847899223624995940231164805338624769792066266598550210732361‘0\
2927‘120.12968458445205 - 6.136223904512810436934385932470010734570603203947 6\
9509976848318933675848885686929955660043047170942022031503588817784042979353621\
111.51704949059241*^-9*I, -0.736700077191917913091293913993714663954337027527\
0995626653153979006121400356025191895580481229197431968725012502435724462596 4\
6269699218‘119.66405157253507 + 2.284837637447242817885592151298358181811924794\
0763598602267015866347465982800276124449267058341913897948337597297597212818 9\
8675111‘112.15561620133268*^-8*I, 4.155888291862233359878109321214313640092331\
8881620675130998814223987609048000349317758087349806705759095474707145863295 71\
2440051246798325‘119.908697356926 21 + 2.014681782372131355303724384211417805\
5524133259329518896713151814746480844519323410622268437432008742929377059585 8\
846218452194 6‘111.594239950869888*^-8*I},
    {-0.682640590224455998777089725655870806682766096925076066374221725297 5\
8857253681272155486413375717583127722546163299539419365003498415094 6‘119.6328\
6932451264 + 1.651513916950254532085204345455107881164437282087387641221048 93\
1030801727712057373281639029225822481246381438834026929066077973 9‘112.0165594\
5413678*^-8*I, 0.4706529894969789901418151919459255610714355379006844747298 21\
5644336088764893629425926667778196570257107559601411183166609429948080239211‘1\
19.836765533602677 + 3.038320681814671420668701204589207793316444586924888 1537\
4968156115852267117780086843119037422200942562241573159914801914466183 83‘111.6\
4669832189495*^-9*I, -2.315021752021078574296740306945876040684385036074274 05\
3154010481475551709387150042739801579225393304728900543590544146710688 1791572\
69133‘120.14325205696666 - 0.215336918822913703411946480984365794609224059436\
4519674588645072468766850668194502853577737313136754286093348067866686 3283476\
892845819‘119.11181547554887*I, 0.439745972098372274992870655570360420154824 0\
7952033269667362139133254397326819911558359184809393117024151968107691046039 5\
561200494745035‘119.84209388201512 + 0.007264448270258544597773881942578 11678\
4723526208158683640965744922592388026528650614213826818781116776485081793 1710\
6188612993135733300 3‘118.06009464736132*I, -0.607407761187540896909595 8 5076538\
1359748916009526740762622151328415616468960257876959168386511143421148929 8391\
7482038585966839537559813 25‘119.593650594578 36 - 0.026588616627341207974442297\
3547047929128993399295221196639621388143836023335264719557036604058299406055 6\
2476430509727273820550536378855 4‘118.234865999855 51*I},
    {2.514990205097050261049118386053268732403182117285160391436572 4954102\
0106819545788814324098773234835518871741472791072728304908350954464‘120.12968\
458369104 - 5.223513645477548811282310343012443840970870688286900329657276552\
0798203108263762068782098841554048698993842482076191343490258 36‘111.447111019\
26123*^-9*I, 0.186345341470664018998489321264004606841328142760085539612 50982\
4018064599238973187472363293351044444291236296431420659851023267400455344 5‘119\
.68290616731474 + 0.215336914593825204888244918925376631504469427255942157276\
0546082981593108415185891062579320027828870969316593790838572198817303695607 9\
2‘119.74570611327978*I, -0.129837900911833417298140182396115259928405101767 50\
0157422792489019829595448629062241040398728964608405710297197761153837802650 4\
07497756‘120.14766061120879 + 1.495547373504966651977207078295520737783738758\
9047314936392303853557071850812859806689631283463758975077685664262999668303 1\
74‘110.20905930002472*^-11*I, 0.217459857624473477176147637634835823047384028\
0594393666262098994573386955274506882041162497461918833453590282046961406108 92\
4270384083235‘119.74227452287305 + 0.207122430778513601701484851721693 1720628\
2461979450574368513234348544839548881584755681312363428385392895868671959 7491\
2253344428740309 27‘119.72112255783394*I, 2.616956062409701420991743504513825 2\
4617802323912259188663491379010934498509064804235156891848413917649747584952 7\
0637953456963067236452 76‘120.12818167420026 + 4.780954328674690344883535166 26\
9454480159338728599235873719777587346329830425701922632424678101583508028919 0\
4041581248093763293 1‘111.38989983812382*^-9*I},
    {-0.736699938080890655738941528437015226209412278696836309019697341620\
2802047125614418825130701597993911875440010857266507343864435067162 3‘119.6640\
5149878013 - 1.493898871556301717711942636176496355510135939708912003918645753\
4404169787012520463590731028091606647040790630322109269859746100 9‘111.9710820\
646393*^-8*I, 0.483476727830949711815928507801456693602111887613248737160633 1\
9226760964219291300706501757158932288652238548350045323670430441670270 4384‘11\
9.881008416405 32 - 0.007264446832048114021896184458139512527436888401472752 47\
2007771780251546339264735116535444404599153912100534114312388724683050498605 2\
29‘118.0578353918076 3‘I, -2.333941598424016834423331081647898883815739327 004\
9263518954358025164900792430917174995941650213841742708465575071010764476297 9\
854641687‘120.14402862111852 - 0.207122426639087084935758240378771322701447 4\
5063213787081726399385130858858938530846145831127930379537250760221723694268 9\
5564738631112‘119.09216576213963*I, 0.462411258179664298958898623812773869361\
1502469596296720014780940506673513197191159487731939566288257101997758277076 3\
4622311705192067812‘119.820559843201 44 - 4.264213535396652674639806030477571 9\
4651519423336928925765962473597172838014507888005845925374195894108509126006\
429908828694411‘111.785370388389 45*^-9*I, -0.830873751985440985157787505328 53\
0541108187743539854855754646836189212011428566407558640040517040673580604672 5\
3573610496304988666096123‘119.71636835807287 + 0.026588615385383372936678137\
9672543749996427818908829245033881120686928500301920771689198471083329989 3858\
```

```
32798043117830583459702785751534‘118.221529040454144*I},
      {4.155888701555051115545675942926583026118134832307743708380846276139935\
0001189950888552777830182631270173097067788866436688632860195106823‘119.90869\
738003211 + 2.756669292049454904985029574643015911663676782089339094168263929\
798673285509908945845504903917145976277205597200773551779975809‘111.73041814\
023314*^-8*I, -0.8308737462215508126224027544739014094040155308979825918622 14\
707181017788920305121236523451459671972961191528555569834878411376901702138‘1\
19.716368631180611 + 0.026588600230787100268579792729696735548954840649528189 7\
087106937466701424438809213579124820667423618311972494736979751962225172175 54\
944‘118.221528799041664*I, 2.616956121831010486374480671067302177237963856456 3\
624395945431188620401991839753127932239111791981156139067828282917868429170 8\
514339793‘120.12818167383291 + 6.224437885410788084721634175106803957859481 16\
105518137215597457719066672266181151691273821296872289593532653776416371106 94\
38716‘111.504485370300449*^-9*I, -0.607407812704482438974792628363674971765487\
906081087424067070948302360928249590205934331107312477143896799665067582845694\
760405844857361‘119.59365062768106 - 0.026588606426495641954548424761179947909\
607698204470458814336299643113066365725683956394826362158962848344044189378 88\
195242999653308570 2‘118.234865800078584*I, 4.155887970911819587472263816160254\
986230747616832070031948263457527407399058013496411030984724458386820072410 03\
030188473977997415571011 7‘119.908697339123 73 + 2.756669134882366434909971185 4\
040879076241400510484770765544381122414523001800019982985014515310784863169 35\
710309113720962088841‘111.730418150917094*^-8*I}},
    {{-0.682640608026945066344664314805183154340580712634344993013410184905 3\
822574868461929210243747551563481966789554465763681327891325149658 46‘119.6328\
6933276035 + 1.795117384008283664648738178422767590558297285588264711489110 14\
390421322608117334974899184866093416743930055967631708597054676548‘112.0527700\
6597952*^-8*I, 0.4706530030225101453875118269368993861114950957613888006434 95\
742190463744778630106510075283816332152703811325993491920049275808241508674‘1\
19.836765595416 4 + 2.119314390138397529861093389798408822432045049772822 29455\
358458103271118324039566372400706069870382136681536368927601610108893 6‘111.49\
026014834418*^-9*I, 0.186345328019851759264441530676753263241012202339 8630135\
666215131233963550720111335816385875297298819566281993240236904667863 92458 50\
731‘119.6829061455458 + 0.215336918685283169812973370618998626914228 4608089033\
999245483404463903991955699310079151304019297104668109495317587225231 53928940\
6692702‘119.74570613111088*I, 0.48347673733628190223836602872283383 5208782139\
055185391378611454167861756234274076359229017991267835566868811330309177315 378\
3908814974777‘119.881008377206 04 - 0.007264452888698987085663471749942 75336266\
761778687246662519061169001083742042691179370284787014891424532214805 88447171\
667068571694298282‘118.057835706157 99*I, -0.830873657333116916278940552 780816\
274430471734523459623738842201681205976211072921789580500115262692799 17116434\
288627562646084838080419 76‘119.716368320888 79 + 0.026588589664437347 5256149257\
393992505147462492519523453170527179333949359739413076939899499169305 11632704\
102231709732414667694517990 69‘118.221528636226215*I},
      {0.470653026584668823886397312232402328213935286563800040057903 27230016\
433644003569891380479539676846320336475688140187695939762569314234 3‘119.83676\
559117276 - 7.386698737557102476110515434867473388504981588429415921 36398747\
118638004809498283429197041098384003546129672864449956043776 76‘112.0325151222\
8333*^-9*I, 0.052028027236033518655606896770048963062908237246802075353907160\
669039743791625899617145617962220012646328492497470328686752400400778049‘119.\
343605390179 69 - 1.98943991974194594752343264023808368232569295687160987391 71\
052866040374507576277231204381432598467437322849580337526089388665 53‘111.9260\
9885942711*^-9*I, 0.166354147871781817572936494648549018752894758060828823500\
232117231426060428930321728618408044510832614337063574091593683906637749957 15\
5‘119.93143749830635 + 1.113655770126862869442035414688436469066404046550210 0\
354732754332660371190794338485308491392018159338272620454956226925747336801‘11\
1.7571548358692*^-9*I, 0.105614224188652342417710243001566454008925718208017 5\
866189634532649812285702859136219074038127815877811803613985859046317062230 8\
787181‘119.66911680506203 + 5.412284013545433397472113419985037879681503585 01\
798596499271468337592966355048589917209693232995096348623929666403376352114106\
4‘111.378774970320 02*^-10*I, 0.46241124083467008830377392684958816137626554239\
767677083003495293240894313358719086715926464378835036564728544302689297601 04\
528629287652‘119.820559847598 + 6.014227305014483960245245148561508499155391 38\
772247504691401819981390392158603118613368033015944782753114441178110880 3103\
812663‘111.934711302756 85*^-9*I}, {-2.315021776974850019914557711667203118416\
559250337906395020342559204611882173385469567462754706453125472207558864014 38\
703535886655338803 6‘120.143252057380 93 - 0.215336915248964509464976832 54777 36\
036356307592828865159176747791612607634303960874590318025491627447209632 3393\
691762001449100298747 22‘119.11181546407389*I, 0.166354136590872828510637 88659\
228323244469817889344826421307251982827868270939390572359389823848134281 49449\
042790144078027429884384672 82‘119.93143749362474 + 1.095644914219607777144 347\
559769199367262684411793736010474911932975764175939787150823665179773 94735 98\
703526202788478220099312‘110.750073716201 54*^-10*I,
      -0.444711155258632461359236040597452425376304275667533036899121 92787979\
852648965361352997913418361288423036201791570521094802492513393424 08‘120.1505\
0950255983 - 8.515418633321717586409722120384632406792241618333572955258 96998\
728751069731737252286988184677962124735701450520520453423958 6‘110.432637482533\
*^-11*I, 0.168479846318544202228370592590003027049869171546179846189184425367\
078710258277408873638577508424831833678179611455861900335858615646‘119.958\
71004632026 - 2.459044730604632626048084586807395524341311412228438524557281 67\
006169038156316244852264859960957605904271568884293078413866 26‘110.122928517\
35681*^-11*I, -2.333941584181926581846619063873942380052561964560786028721 17\
421355240333267190209298968707917936905512501554448476097511953671436515163‘1\
20.14402862089503 - 0.207122430065109213620202895964016868875735454287286384 9\
158056833023369336898130588013177493778496036379450448596523643568823085301 28\
48‘119.092165771749 93*I}, {0.43974596407558259600977881983695004995373022055 7\
460050855807282237296886321203593304252588682576440817444615103325951666017 71\
2446061974‘119.84209385747 12 + 0.007264450871271847294539960735964400904789 01\
634214718891062565354625676406413694502850736822839533152140795612508327734 12\
06833790912289‘118.06009478623851*I, 0.105614232526475726144729231623810654 93\
```

```
396441436241635094014294308283949856816253127505407190266855440669181935046024\
19570805848830821975'119.66911683909366 + 6.040259986036811029864894641485293\
68657648467881202646527277966307178897459856586945870480541718934406589015644\
96206666088498'111.42645002363344*^-10*I, 0.168479834794505765603890436281549\
18191890305441267012119653341707807996962764980950015305282071903606403732054\
30219194125571591318754411'119.958710043630776 - 9.231968222380178563583701157\
49552106219265673428103819054341027637026333840157663656834546328392580169708\
49744747443117704311'111.697456416560144^-10*I, 0.10561421660184883943425859\
13399933812274410789309290194652607456784915807316056894150976510622000467590\
57156432634084019017400349676'119.66911677985676 + 6.040259469602741679007654\
82293175347075823409413885745526363473903627024189436442541691654189638664437\
33863674472427471052541 2'111.426449997284833*^-10*I, 0.48347673563694736465506\
97746741292208824066513635388032821080957828751486920589692820387006187366146\
13357285198903424795135658042096684'119.8810083981242 5 -
        0.007264449368680261379079896545955082314965036762221576640313833969 1\
83753497380692910085373035328924141107806905677633048736359798590059'118.0578\
3551816349*I}, {-0.607407894326608189045673673329780331141992175765700094900 3\
52503923154417331815812585580412910847026247291053009636262803882218189873207'\
119.59365069471396 - 0.026588590876922440652061219968274622360060391807256262\
94513589876504653849523230762032225905222074413653446314424047638111224019528\
5198'118.23486558419347*I, 0.46241126661987118068910735089003242429007442570 3\
84183383660672876286374418196569094806529590921156672462125551501522882117106\
86538138'119.82055989657688 - 3.151678412613061070438826380255856844825802834\
65073313091098771765979941820529432814081435688909254740519488737955002308616\
0167'111.654073386509688^-9*I, 0.217459873480535205475066470896569742215548 0\
66833484860604317966274784927751339209001464851100424255492063168077226730346\
42364737129237'119.74227456301465 + 0.207122426190195737787901148241114491502\
46413445283548701511055939885322019150123521074169772195977288091300941090591\
78270761729101121 46'119.72112255668882*I, 0.43974594211677150646908850349311 91\
64252332745338675947670097881212822427231416458667241483681911181427393994975 1\
13952489313279181943809'119.84209380192215 + 0.007264454520439049542046444796\
17347720313277032097644555592012731612975854962477764797423441131464086733453\
41021286087721044499474117734'118.06009497053606*I,
       -0.736699865696433092978031693271690466008845339877154665528233321384\
67459670433843891227784588624698674176886057461213489038638512313045'119.6640\
5145082725 - 1.634555059046286169485154494675114109739760052206321365037193 02\
15621796037719720599370462619198219907899771609070472869353281564'112.0101604\
1426509*^-8*I}}, {{2.514990128168997713698787918820431987366273208718606196 58\
41383813730886145267791008140240780622368591975695858789658853495774378924558\
29'120.12968458372102 - 5.345755349485051423990478311675611839768701322710155\
22484828133111851212202646551693513972308343847432888905103506377565702038 28'1\
11.457157378241 13*^-9*I, -2.315021743571800568695368727272801950353769269597 4\
86183057281885773242140200312278800215691528773687380096547533537731221055006\
068447923'120.14325205676948 - 0.215336922471482816273379895958362590621116 67\
45561748751059799785335457893340289272611896259130304047553447750747023414084\
89360156945355'119.111815484295 26*I, -0.12983789860285198612489233979968931 9\
99226329191653009752983386301193202177463159406668600648218160338924940063371\
45652585319540831747 4'120.14766063639154 + 2.039875948549875603360426530297 99\
01549750036246980846113590832522562633191856073431740316688762847172216732122\
83649249496076 39'111.34386291585254*^-10*I, -2.333941608580828372291328208707\
16299940367331952349853607938888822850581039530492004143452393078729979401152\
49330464862633720680528398'120.144028621335 86 - 0.207122422931411692762611 17\
09568664323623527190102727779842516674973402103836352189077505303982113220933\
47053245448679172899216236508135'119.09216575269276*I,
       2.616956144923212147301834623241958858720747241213108071304246292001 95\
56427805744455714634721367319289951736611874946995739998557706714573'120.12818\
167446073 + 4.658712778078843321713174156736056155758946894185712321887838889\
11850288535121162382940027612770042437346356963995449910049 7098'111.378651165\
56182*^-9*I}, {0.18634531572830257731129717000167754786216686134001635620 45\
73\
15506116540197167928688637291234126814523699067733727334252350345762720409'1\
19.682906112162 12 + 0.215336921723248022162055878480252580382755728119808 5307\
17675091670146278948372062609131601647377274922032455278478985195666236739684\
653'119.74570613250074*I, 0.16635414694549571803233475341042988902701994777 11\
44760940866615733217560451126836717369698830750599626179800074980661622025 15\
003566402'119.931437481880 88 + 1.134580132814179445738181776685985481714448 11\
56019915236657801210637279779846249213804125576250067012563533403725109908477\
0352'110.765239024802 92*^-10*I, -0.444711153628559843595250114717962014389644\
51849667403853198006323262814276277889278443257521624165755475728037598761996\
80790500705362 51'120.15050950256594 - 1.052115755728614926443647903002616203 6\
44259213066963745262482428898163577292366916026461090932573384737701538288411\
41160532514'110.52449500493994*^-10*I, 0.168479833820322958975816921409205 186\
40228984809913363804274594419769454585272431549538751724389180616304515433241\
75231376435943878999475'119.9587100541445 - 2.069695277791623943686966642559 35\
22214681745375012443183965140653387029198881853004456400555331472749158057104\
592016282828874'110.04806853745104*^-11*I, 0.217459889997323431065641390531 82\
23448496080665933295037397778153515973651378677048342686441750589015724676990\
12441092321036391291250379'119.74227459090967 + 0.207122423551720614411511645\
10086400235325227192255350398011958855419273995360174086130958662822132185684\
85139355640483492371733472725 59'119.72112254606479*I},
      {-0.129837901439901419173238631691092885989278126089524399958680738704 2\
08464449343857315371629862219860801157726055809071291934706913593935'120.1476\
6063196038 + 5.211911086621925481178417875302958511247208022514901176988529 58\
20990449581362149979062066015039304918748372648762626320239354'110.7512561427\
4603*^-11*I, -0.444711153940592895255599257139028299773962082844094450202122 0\
20570096136041060735286527846903628859210099138045156242848142649385744733'12\
0.15050950256582 - 4.312395099261535478035235222703188110764823559230758473555\
81965002722803673981297627606300809366604389445298472497227242955'110.1371\
5002412609*^-11*I, -0.305217935458132264758274524847122038599125406274921358 8\
27142842015173766556817005120492574170136064291774464971634227166426939797742\
```

```
392‘120.15051499783199 + 1.12285892309902758180559609183517615196832202270087\
4749195867218791700038329087957591421556792002860912264983309260399406737096‘1\
10.7162301420228*^-10*I, -0.4447111539405928658652124221566591148650909293760\
5253933766173316408616435787255296041353030911687183085305705652387525995281l\
138592125‘120.15050950256582 - 4.31239508144097068962392755042675527100098716\
3562882806924535478568170115698422055788708961234908731182340883512204808638\
21‘110.1371500223314*^-11*I, -0.12983790143990136172812035716644127265181715O\
0998401979400459378587639342507017754607993820215407909354707105469848975277\
6041200545594‘120.147660631960 39 + 5.2119075811831079108135929145505046835225\
1992791137864482163206228425645686560128612923249914556414350572754904715862 9\
5250462‘110.75125585064718*^-11*I}, {0.21745988999732341136847434143136873622\
8383160221230614619492994740975373227240103226587278401297194532338469698676851
1867281923208155339 5‘119.742274590909 69 + 0.207122423551720564224603096374972\
2815785522194218814940070742953832661536829758935392265069928784156337559600 8\
9282358197570227158160 62‘119.7211225460648*I, 0.168479833820322940186104331381\
2982320209857500756005178297660406827627602839214997697069351353754690761748 6\
5880019119364586413997581899 3‘119.9587100541445 - 2.069695685264708224218456 6\
4578530222832866039920604060743961271645645315648438065747333052327620455507 19\
1342085788424153036896‘110.048068622953 14*^-11*I,
       -0.4447111536285598822859629178183685143559592103397249035326592737774\
6803634642144833536159528982366003848183828960334778763518049141957‘120.1505\
0950256594 - 1.052115754330655275652962816792258164939382136750098911977021 43\
1482990573279083153564150934511487573767069050525639200161619 92‘110.524495004\
36288*^-10*I, 0.16635414694549569389130108062880438225582821514939341501735 75\
9620250344556900063528523646303015330420466605573138573710061529087693411 5‘11\
9.93143748188086 + 1.134580423857711864919819010446229379478455289131181737492\
1105662481644970531964500437938382106607278569898030405824536243555 83‘110.765\
2391362085*^-10*I, 0.18634531572830253586484987610553346302642442060022751851\
5920423956857755712861506814433819934794271037026356313928959061796754532464 4\
96‘119.68290611216213 + 0.2153369217232480166350126173684145079272487128514 67\
2910621416958516211542571706614851656133753463727597655406104317164197102958 7\
7869108‘119.74570613250076*I}, {2.6169561449232119904145496725015966156173344\
4116954898837943230565105414190216508801001633541989098569637030821606353367 4\
1307425891039 4‘120.12818167446073 + 4.6587129531898187009980414416109811748 2\
5745966395381082509049033932318800328831464494072504826284134683104402021281\
25695677293‘111.378651181886*^-9*I, -2.3339416085808284139651979913816651560 5\
4054534546679357851203552183380695891525957412180601095069814806595511074280 9\
6200584154219049919‘120.14402862133585 - 0.2071224229314115952446887270156469\
3463224825983551280754185705359936692153392830600482576143849332015985409 11\
7712747339624514527980 87‘119.09216575269274*I,
       -0.1298378986028519772521583639556144210937525539008908822670851807782\
2291564855880138095403012078689221614534383995481487777489299504009 3‘120.1476\
6063639154 + 2.039875959031511822962014269199677669916383809614064925110283 79\
9185653168037280478598967015559005897560050163367271369315738 16‘111.343862918\
08411*^-10*I, -2.3150217435718005776005484889216798105586288924613156375876 59\
9041878521991718349127823276898506090588747925600888869156480773413096110 3‘12\
0.1432520567695 - 0.2153369224714828582192922995004394471436560139465879222117\
5895520483780652496970631937839714676038970320305461751830473611926166216528 4‘\
119.11181548429528*I, 2.5149901281689979630956446195910072285791472620985746 1\
5089632013900410679387206118395002903848994561047150654988905424208714875602 9\
38827‘120.129684583721 - 5.345755538252449444719402649068334411931158547731 65\
3856046322819542544703677942329917281969967626007039238566061166182174796855‘\
111.4571573935767 6*^-9*I}}, {{-0.7366998656946325617402442245828236823121217426\
7197114239669244500768470067598500866936174868805930831398977047617543843075 9\
5053842511164 8‘119.66405145082277 - 1.6345550793681823715310361333787635870 06\
8946829128833519690519499524130854900885058866947546045248448972108776862519 62\
33003280556‘112.01016041966456*^-8*I, 0.4397459421167714901403322239957197891\
0602776484462539465497547728638402660530034216132618365262019640457510562930 9\
8898184699653239424 27‘119.84209380192215 + 0.0072644545204391777826089062229 7\
2292243274500604807346692219627311064178801772388648185967346244305036441860 7\
0516583960745460708427756 6‘118.06009497053608*I, 0.2174598734805350987477414 3\
0197762501520888714042518294088588555356466505489523880142563690279554788542 4\
0113188569676781226806459159940 9‘119.74227456301465 + 0.2071224261901957186 09\
4627979927024364561739797258866079407216730420088690722642937942940560174932 12\
1479805788078981084031264010464956102‘119.7211225566882*I,
       0.4624112666198710016439120519566181392627234501754385435060204858150 0\
6709207395882169619374186587493914190275814794172466072299894468963‘119.82055\
989657687 - 3.151678252013033640067378431639314010587068151810059345812429673\
5962324715006000132411383196264502810323608039392245574580125 77‘111.6547033364\
37936*^-9*I, -0.607407894326608232544941885749850516044727339172288052226357 12\
5726904977976064461495111620665567268998668920548972499572692348649912307 4‘11\
9.59365069471396 - 0.026588590876922421186350884594866108116539406171457694 70\
0443701416919630590410715795195637478149470671736461730965362993906505090 6392\
47‘118.23486558419347*I}, {0.4834767356369474555481695568922695981043153043 70\
3328562619035874987919470836645125975387295216911319122094797831701157287094\
4220751518‘119.88100839812425 - 0.0072644493686803812010237796207539896312 76\
2050836398340120692621802930445367647956784881659435108814272101887436736 2982\
5639618590674‘118.05783551816349*I, 0.10561426601848802264075604261561899 63\
3922791279421836911482838961202831206600197520325304374388065459148351049 9154\
4022593333265361670 9‘119.669116779856 76 + 6.040259781023470696762451465985 786\
6667159250397752271816740942983079946473077478430726080753053375826276667 2315\
8547102481145 57‘111.42645001513947*^-10*I, 0.16847983479450571775239304350 118\
0576731288955455194780803835969356766931397417285754054556456434667652955 969\
7946486988977484135188016 5‘119.958710043630 76 - 9.2319678581345093001602 3851 7\
8746597722034526427813023533741861717873049573092100694728547355596968748 9466\
0721471482936427267 68‘111.697456399425 12*^-10*I, 0.10561423252647559768656 235\
5042957781815247768636210864302248164549624515703809128852583526070597610 447\
6727402540662934915835998729227 7‘119.669116839093 65 + 6.040260038293767166 857\
1617609546635251763040645403301938666444263629328711202449373888151054853 8548\
```

```
669273334080706022613641449'111.426450027390696*^-10*I,
       0.43974596407558275145158871368483749022278593212330473541179870465917\
774417525744791840709541490391958445925179838643892745966630005244 2'119.84209\
385747118 + 0.007264450871271841157267709648463283360513315109306765023105 32\
288119754779298161618126571074154784903660771933957693653649077340 2828'118.06\
00947862385*I}, {-2.333941584181926777792085968561292542973324462549924120422\
175278398514980246968498562491778164457521389309041276551405486116739471403 13\
3'120.14402866209501 - 0.207122430065109296984077291334287386216118357243377 9\
655694717761543132220131124539807520999658582902208234083150669336824889380 74\
69608'119.09216577174992*I, 0.168479846318544109137192209611581965200533609 19\
251041759838124755100572023328649780599630538233244980637958091552319430 5498\
9096659449'119.95871004632026 - 2.459047463451819454780874036880426121063182 0\
896468116496824415154995718526236179788730331668830841097259435797315960015 82\
8944'110.12292900000756*^-11*I, -0.444711155258632658484155921641083592546480\
174686676041471338544093769536389662120552552761503990343798454861972491194 13\
8042227876226777'120.15050950255983 - 8.515417278855267037276019062134866035 5\
858944557512286039040133453911978658158906317502554288290068052318292359969 37\
3266816549'110.432637413453 93*^-11*I, 0.16635413659087268017629546067043150 98\
407080530973024430676092103266784958500012796740497415768239822253568939021 59\
012650992963822158848'119.931437493624 73 + 1.095645326690144155690822013110 14\
185144902376500443617457878547213804990501933940620203210342226645252581 1084\
69623095926360 56'110.7500738796976*^-10*I, -2.3150217769748503213567444203266\
196559427418658520098402150883162196108033661110218244723412183499149940067 53\
691022274400079597939160 71'120.14325205738093 - 0.215336915248964483784335283\
825777230671703668908443952355146562552388955087414382009978072313649703279 94\
475503555697951334448791930 2988'119.11181546407389*I},
      {0.46241124083467006427868157218150897216016339046542465817483914673 2\
822736180172829573540066397499590029200559563216781057326372509390 5'119.82055\
9847598 + 6.014227237561471192976321783837659258858994324260759533812712153 75\
645555638599293708745346810992378588802587955698353616551415 2'111.934711297886\
*^-9*I, 0.105614224188652159385436653589439186077977751502491245880364 2177217\
830718267337549092802314580338671472329996408550027089977648732001 61'119.6691\
1680506203 + 5.412283910550222898241985607770437557175104680985628191 5230692\
964175321922495963896758593140706538160515021520067448119198586'111.378774962\
05545*^-10*I, 0.166354147871781615328702887542958376935620197826534470528688 8\
366374612676587728853966978095329348516536606995853085251244827979798369 82'11\
9.93143749830635 + 1.113655756130029651632082263859555269074389388188764 31161\
588063398904239879879894691042026582161799777305822781371329231446447 7'111.75\
715483041083*^-9*I, 0.052028027236033344060111833043898935145864295842152 33 14\
891031870644851610331570304236140653523358068529121204503359043804678886 05612\
82'119.34360539017969 - 1.98943996011853656168972188248810077050901218291 0596\
990490254228582158496146075558762483116619587785353012239560989512386768 46'11\
1.926098868243 33*^-9*I, 0.470653026584668865887377871200512436862421106731 4115\
183775407594320788142550370083399498106225389196518659647034445969793583 46337\
0334246'119.83676559117276 - 7.386698694511038564932310090241822873545 9447923\
691606464765479159340828263023964596976347929891347606873890970286235753 28683\
666'112.032515119757247*^-9*I}, {-0.830873657333116366768307070672066115300 532\
427060348910203591830528641643184436717902535614939200757339763282958245 96303\
2348564101434762'119.71636832088876 + 0.026588589664437202900345821442353 6333\
970969658365542586125440173649672497056911433552302290371457828983005726 93105\
876371206534433102564'118.22152863262214*I, 0.483476737336281762473481136 67750\
905557449167380964404068893088782069128749444049130242452412658350424798 33578\
266150264697300133565655 92'119.88100837720606 - 0.007264452886998154747 3905\
783182582250603645579482492723242779452754458035638200293375467232637 09992287\
983509839723937158578424651918 73'118.057835706158*I, 0.186345328019851659 1139\
578537401383296877246879114089072099471873336851196134021550488591779076 09257\
696584629338976849973754529931409729'119.682906145545 79 +
       0.215336918685283157102707846476320500602838157008189103581930 1584688\
144658933787245179887961654568465826738230354927882648578088601327 48'119.7457\
0613111086*I, 0.470653003022509924780422696638107213326774047259802249452 7790\
001522383361629435494176638159233702213372308949928495575740094199566492 79'11\
9.836765595416 37 + 2.119314471085213301365162247337300087451549874562393492 296\
262971411136477974633455800357617464753836166226496052685317051158308 7'111.49\
026016493197*^-9*I, -0.682640608026945095259031556775787896807290279511 4761\
292225367249668257375226971329542758796298052061303343967498318339708909 33669\
6712'119.632869332760 35 + 1.795117389853315122775416795467439413431045292 4103\
517440325658447874804970643756155040254711566947195837132143971304282958 16846\
3'112.052770067393 62*^-8*I}}, {{4.155887970911819760212885071394739748624 93851\
367016647813403144403218089062649732054700001083454188934249679731172694 51568\
54157442225723'119.9086973391237 + 2.756669134241974060585838447298303179 6832\
799028904361104669164935013970759042424430949526643862671385728542799625 70457\
3025371676'111.73041815081619*^-8*I, -0.607407812704482343964029014653460 3092\
904035143631961252864240067674999765357996567556906699442690780800745154 59774\
224820754333283828248397'119.59365062768104 - 0.026588604624956404593940 1719651\
237042528007711759206257044539323164854082356692045152169232385318224272 02244\
030264564798031790613785 19'118.234865800078 57*I, 2.616956121831010114623 18132\
071019680632180962415641879423944611822506145257498561766191863338592391 01030\
806594823979593352890415646132 3'120.1281816738329 + 6.224437880921292422 31532\
949487807986309609222076289667754554620639146967630129685358889270644095 20306\
747318270965503872482400 37'111.50448536998726*^-9*I,
      -0.830873746221550878167816524486367212946549833625527532027552766 72\
015167866140380112467389680514583987584527463828767303974634701931 57'119.7163\
6836118061 + 0.026588600230787278448563978412461178843670439979765572263 48595\
268074978414786074830155518693645768080137299230995930653396856215787 825'118\
.221528790941 66*I, 4.155888701555051703288464351314675315846630897291 5881848\
243404015444028025987088230498509597219408236329426748512038271167550 8936157\
22'119.90869738003211 + 2.756669288688439084186806404513658012331804 160762172\
334506418971465864219777447861989365688667458229190240505820571957 851898959'\
111.73041813970363*^-8*I}, {-0.830873751985441101465119011130888976763503730 2\
```

```
5473711968331958795949409172232034840373275327901009478831676896388690194754 8\
781024424798‘119.71636835807288 + 0.026588615385383259362546891999432477341145 4\
156160032393017875937125954925100986196785567515095536122730076192121903804985 \
6804492618730152 9‘118.22152904045416*I, 0.46241125817966433197563490829347977\
313302402854265506892083324623407982134522541296634287293820963074685586605925 \
428563106891941278302 01‘119.82055984320144 - 4.26421354329035645400582687488 6\
937858826285242390551463373098809324458995182106125498929026943452325235607985 \
306062034243366048‘111.78537038919344*^-9*I, -2.3339415984240170305892791536 9\
207816843547599666914653414636994390263130863716028359833541434702938330778 56\
581277992972382532759274420 45‘120.14402862111852 - 0.207122426639087057674948 \
371622167893646482391813341524683549675577989037490625582735507070402305005 57\
853965416837494083307174573454774 03‘119.09216576213962*I,
       0.48347672783094970473136901807187288451013635670261468660515495596266 \
290421099912632474466156437257824420879876783430190806551641521263 1‘119.88100\
841640532 - 0.007264446832048237812318712107679612483282672018412585168342031\
645551858197316025848628982174176567814898001135117267232871715544497469‘118.\
057835391807644*I, -0.736699938080890652898076871866532002501961183018265402 47\
271083066778165192650624482522447368597718228040983470860918479982981384541 62\
87‘119.66405149878013 - 1.49389888871815614839910175371607053253742980657768 0\
999492851595216406967413795853539242279208237821785717215455628027947885827 7‘\
111.97108206962847*^-8*I}, {2.616956062409700809480461579709302041187285526 15\
664623860857815397023174337599609895007893741349752979852013157649343786642 8\
343832372581‘120.12818167420025 + 4.780954476353624504315944456344353007539610\
748337545965176824766097833285175092308575716161100761334698621150323263925 19\
5877685‘111.38989985153876*^-9*I, 0.21745985762447324937122715647907255599523\
420497534816727599302559839035489607522384862645467600054559686280370134181 47\
527394737209652 56‘119.74227452287303 + 0.207122430778513595748888401840116187\
074612887438235227033414854154710075888442294594729550811720396121123057187 72\
870796468670112717659‘119.721122557833992*I, -0.12983790091183373344233620 5422\
012650554431706260608808870784223593136910629648072692141781401680057280 67047\
979507137669643315147744821‘120.14766061120879 + 1.495548245342711183602 6250\
305802424068627237675832696014761966890034370204644587298255338432771745 91664\
406232079449432465124 2‘110.20905955319901*^-11*I, 0.186345341470663804579409 4\
310197677221093122447340851838337783242078507587444623779211312049805722 51397\
209100382781473454404146777793975‘119.682906167314 75 + 0.215336914593825 08493\
273495472945254535352898453024224770763879047444945482665976357868496658927 97\
810637364735425542101613780479054975 43‘119.74570611327 98*I,
       2.514990205097049611204413064523094970015467844024611180303922781535 23\
704810170389560175816265571644272196900670817019248569769458927949621 20.12968\
458369103 - 5.223513408670544403470587550895532763699263677241961363268741708\
731874490202971248606033294464308837323450982788193948923599763‘111.447110999\
57257*^-9*I}, {-0.607407761187541020884335677148993474329512031727300927705 95\
236619207127243893978595474980569751019823283899815672567600966762680173385‘1\
19.59365059457835 - 0.026588616627341367134933562802279231192617812745693057 7\
235112610569131031269340712808500668357637058782566833860042467429663828454 47\
12‘118.23486599985551*I, 0.439745972098371947541463361422182432057459559377 70\
421444885573156101992282378172699340178972088387048119825769411373758659138 18\
30067411‘119.8420938820151 + 0.007264448270258578113131347640634577695341063 3\
926901023777339289291998937436133265617298875434191166016819166793729480846 16\
60882302823‘118.060094647361 31*I, -2.315021752021078756396883796412162060 4165\
962489579958327291896543026794804716939053230665956840420900513125141534 53180\
048713785773710649‘120.14325205696667 - 0.215336918822913726019548812962070 58\
765790841485406847692175207290168947628639676225555481290677058062295493 92615\
471701162855825879282 82‘119.11181547554888*I, 0.470652989496978733759114716 53\
738574626453219513257179300878314343554833524655234855343614430728933685 92314\
476726296916283015954830774 22‘119.83676553360269 + 3.038320600174082543934 449\
424333244656732405641311525704075820052819203004618082544369975685954444 20319\
669667309225250496829317 5‘111.64669831022535*^-9*I,
       -0.682640590224456008511077442722458143238188320530531721483182238491 4\
509715114807116580431657077173708602282577433693119086619499443774 4‘119.63286\
932451264 + 1.651513893139569391308653703875929803168278501133312021681567181\
470779052841099924698850693180007624172835197974106427580394240 9‘112.01655944\
787535*^-8*I}, {4.155888291862233173111771076029301166690368491582926513453 52\
076063171865867574788346184311956897000511420389950913354472091087995950693 61\
19.90869735692618 + 2.014681800740959447545590393673190510425555342930612012 52\
798216587413529215456351558357574106759449893773026277259274824752480248‘111.\
594239954829 55*^-8*I, -0.736700077191918384887328499004469933698184244908172 \
342477102111699990030825402561027731387940048661739751654728611724223643259 57\
933524‘119.664051572535 07 + 2.284837629204379765326726439648758632835418366 2\
664997721311326123659777593862648095517043723947374546513468957795588960798 381\
378‘112.155616199765 92*^-8*I, 2.514990170707305461358853053286661912277391048\
496953405245973292833642499496952300911525203167181126085251183037410381835 70\
1271285167207‘120.129684584452 06 - 6.136223947878966021906457069182196 91996197\
443902854496696656382021108201036184506064703669638241332143427769406569739 16\
138110357‘111.51704949366 17*^-9*I, -0.682640420459842322297718817234718709054\
978993196255511048041986849575039546501014130554932546398395929169628488477 77\
257191071712030110 2‘119.63286922678103 - 1.942938496036365194364469107358976 7\
000199172409382119731353387823542157860729318177156629116164487327148416420 80\
9741043960355024‘112.0871362798313*^-8*I, 3.598731026324083744789969454489217\
723175921790496685764651448606719104479128657221546112651571252606534890695 \
666295867392901351033202 2‘119.870197984857 46 - 7.341425820155356213020419014 2\
186448585872408840816846218976493761433321377664904109939126918222910458083 45\
686007455708290951953 3‘112.17982901162122*^-8*I}}}}}}}
```

References

1. G. Baumann, F. Stenger, Fractional calculus and Sinc methods. Fract. Calc. Appl. Anal. **14**, 568–622 (2011)
2. G. Baumann, F. Stenger, Fractional adsorption diffusion. Fract. Calc. Appl. Anal. **16**, 737–764 (2013)
3. C. Lubich, Convolution quadrature and discretized operational calculus. I. Numer. Math. **52**, 129–145 (1988)
4. C. Lubich, Convolution quadrature and discretized operational calculus. II. Numer. Math. **52**, 413–425 (1988)
5. F. Stenger, *Numerical Methods Based on Sinc and Analytic Functions* (Springer, New York, 1993)
6. F. Stenger, Collocating convolutions. Math. Comput. **64**, 211–235 (1995)
7. F. Stenger, *Handbook of Sinc Numerical Methods*. Chapman and Hall/CRC Numerical Analysis and Scientific Computation Series (CRC Press, Boca Raton, 2011)
8. F. Stenger, H.A.M. El-Sharkawy, G. Baumann, The Lebesgue constant for Sinc approximations, in *New Perspectives on Approximation and Sampling Theory – Festschrift in the Honor of Paul Butzer's 85th Birthday*, ed. by A. Zayed, G. Schmeisser (Birkhaeuser, Basel, 2014), pp. 319–335

Index

A
Abel integral equations, 43
Abel integrals, 43
Absolute error, 56, 70
Accurate approximations, 40–43
Accurate error bounds, v
Algebraic decay, 99–101
Algebraic equations, 44
Algorithms, v, vii, 45, 47–51, 54–56, 58, 63, 75, 79, 80, 84, 85, 88, 89, 91, 108, 110, 119, 120, 124, 127, 129
Analytic, v, vi, 9–18, 20–23, 28, 37, 43, 54, 61, 64, 70, 80, 103, 108, 109, 111, 119, 124, 129, 139, 142, 143, 147
  approximations, 37, 43, 70, 143, 147
  and bounded, 20–22
  functions, v, vi, 9–18, 61, 80
  structure, 54
Analyticity, 22, 28
Analyticity property, 9
Approximation, v, 7, 12, 37–48, 61–66, 68, 70, 71, 73, 75, 77, 103, 104, 107, 111, 119, 120, 129, 141–144, 147
Approximation operations, 38–45
Arbitrary accuracy, 42–44, 107
Assumption, 4, 5, 11, 14, 16, 17, 27, 29, 30, 80
Asymptotic decays, 72
Asymptotic equality, 28
Asymptotic property, 9
Asymptotic relation, 22, 25
Available kernels, 129

B
Banach space of functions, 19
Banach spaces, 19–20, 33, 34
Bases, 37–45
Basis functions, 37, 38, 40, 79, 103–107
Bell-shaped, 56, 60, 61
Bell-shaped function, 58, 59, 61
Big matrices, 55
Bilinear forms, 28–31
Bilinear operations, 28
Boundary points, 99–101, 103, 121
Boundary values, 54, 68, 99, 141
Bounded, 2, 5, 10, 13, 14, 20–22, 26
Bounds, v, 15, 17, 22, 26, 39, 81

C
Cartesian, 6
  coordinates, 80, 81, 83, 85
  product, 85
Chained sums, 89
Characteristic time, 54
Class, vi, 2, 26, 72
Class A, vi, 72
Class of functions, 10
Collocation, 60, 111, 113, 120, 130
Complementary coordinates, 67, 69–71, 76
Complex argument, 108
Complexity, 55
Complex plane, 37, 41, 108
Complex-valued, 22, 26, 144
Component, 3, 4, 21–23, 29, 30, 34, 46, 48, 56, 58, 59, 62, 64, 66–70, 72, 74–76, 80–83, 85–87, 117, 124, 136, 138, 142, 144, 147
Computer algebra programs, 53

© Springer International Publishing AG 2016

F. Stenger et al., *Navier–Stokes Equations on* $\mathbb{R}^3 \times [0, T]$,
DOI 10.1007/978-3-319-27526-0

Conformal maps, 55, 67, 79, 84, 85, 99–104, 106, 111
Conjecture, 40, 42
Contour plot, 61
Contraction condition, 56
Contraction mapping principle, 33–35
Converge, v, 34, 63
Convergence of iteration, 33–35, 63
Convergence test, 51
Convolution, 41, 43–47, 49, 50, 53–77, 79, 82, 83, 88, 92–97, 107, 110–130, 133, 135, 136, 138, 141
  in Fourier space, 92, 133
  integrals, 55, 60, 61, 77, 79, 82, 83, 88, 92, 97, 107, 110–129
  kernel, 107
  procedure, 43–45
Corollary, 14, 31, 74, 75, 77
Curl, 57, 58, 80

D
3-D arrays, 48
4-D arrays, 49, 50
Decreasing, v, vi, 15, 26, 57, 72
Delta function, 2
Denseness, v, vi, 11–18
  of Sinc approximation, 12–18
  of the space, 18
Dependent variables, 53, 103, 129
Determinant, 26
DFT-Sinc, 55
DFT-Sinc convolution, 54, 55
Diagonal matrix, 42, 49, 50
Differential equation, v, 1, 3, 44
Differentiation, 4, 28, 41
Dirichlet, 67, 73, 77
Dirichlet boundary, 67, 73, 77
Discrete Fourier transforms (DFTs), 56, 79, 86, 88–92, 133, 135–139
  approximation, 38–39, 45
  coefficients, 92
  matrices, 39
  matrix, 48
Discrete matrices, 89, 92, 107, 124
Discrete points, 84, 99
Discrete representation, 64, 85, 86, 107, 112, 121, 124, 127, 132, 144
Distinct points, 37
Div, vi, 1, 58, 80, 81
Divergence-free, vii, 3–5, 19–21, 23, 30, 34, 56, 58, 80–82, 117, 129
  condition, vi
  velocity, v, vi, 1
Domain, vi, 9, 11, 55, 56, 58, 60, 62, 63, 65–69, 74–77, 81, 82, 86, 99, 100, 105, 108, 120, 121, 124, 129
Domain types, 99

E
Eigenvalues, 40–42, 48, 107, 108, 114, 121, 123, 127, 130, 131
Eigenvalues of Sinc approximation matrices, 40
Eigenvector matrix, 48, 107
Eigenvectors, 40, 42, 48, 107, 114, 121, 123, 127, 130, 131
Ellipsoids, 58, 60, 70
End-point, 41, 43, 84, 86, 87, 99, 104
Equi-spaced, 39, 46
Error bounds, v, 9, 39
Error components, 64
Error decay, 56, 70, 74
Error margin, 63, 65, 141
Existence of a solution, v, 6, 11, 19
Existence proof, v
Expansion coefficients, 106, 107, 142, 144, 147
Explicit example, vii
Exponential convergence, 99
Eye-shaped, 10

F
Fefferman, v, 1
FFT, 38, 43
Finite, 20, 37, 41, 54, 67, 77, 84, 86, 100, 110, 121
  difference, 55
  element methods, 55
  interval, 45, 84, 99, 100, 102
Fixed point scenario, 56
Flow fields, 56, 62
Fluid, 53, 54, 58, 61, 66, 68, 129
Fluid-and aerodynamics, 54
Fluid dynamical system, 54
Four-dimensional arrays, 48
Fourier amplitudes, 91
Fourier polynomials, 37
Fourier space, 92, 133, 135–138
Fourier transforms, 5, 10, 13, 14, 20–22, 38, 39, 44–48, 50, 51, 55, 79, 88–92, 132–134, 136, 137
Frobenius norm, 139
Frobenius norm of matrices, 139, 142, 147
Functions, v, 2, 9, 19, 33, 37, 54, 79, 147

G
Galerkin scheme, 55
Galilean invariance, 53
Galilean transformations, 53
Gauge invariant, 80
Gaussian, 60, 72
Gaussian multiplier, 56–72, 81, 84
Gauss–Seidel, 6, 7, 63, 120, 129
  algorithm, 120
  iteration, 7, 63, 129
  scheme, 7
  type method, 6
Gauss–Seidel-like scheme, 7
Generation of Sinc points, 79, 84
Generator, 72, 103
Gradient, vi, 4, 82, 135–138
Graphics grid, 142
Green's functions, 2, 20, 45, 47, 60, 61, 79, 82, 83, 92, 108–111, 113, 119, 121, 124, 127
Group parameter, 54

H
Hadamard products, 43, 47–50, 128
Heat equation, 2
Hilbert transforms, 43–44
HPC computations, 129
Hybrid calculations, 55
Hyperbolic multiplier, 72–77
Hypergeometric function, 24

I
Identity, 13, 15, 20, 22, 24, 27, 30, 43, 44, 48
Imaginary part, 26, 144
Implementation, 55, 56, 59–61, 63, 79–145
Indefinite convolution, 41–45, 47, 110
Indefinite convolution integrals, 110, 111
Indefinite integrals, 43, 79, 97, 111
Indefinite integration, 40, 41, 48
  matrices, 41, 48
  operators, 40, 41, 48
Independent variables, 53, 103, 129
Infinite, vi, 13, 110, 121
  continuum, 54
  intervals, 37, 99–101, 105
Infinitely differentiable, v, vi, 2
Infinitely differentiable functions, v, 2
Initial conditions, vi, vii, 1, 9, 55, 56, 58–60, 62, 63, 66, 71, 72, 74, 80, 85, 91, 92, 117, 129
Initial flow, 58
Initialization process, 79
Initial mesh, 85
Initial pressure, 58, 60, 61, 69, 83, 117, 119
Initial value problem, 2
Initial vector, 33, 80
Initial vector field, 56–59, 62, 63, 70, 72, 73, 80, 81, 83, 85, 86, 129
Integral, v, 2, 15, 23, 41, 54, 79
Integral equations (IE), v, vi, 1–7, 37, 41, 43–45, 51, 54, 63, 97, 129
Integral expression, 4
Integration by parts, 23
Interpolating, 37, 38, 42
Interpolation points, 41, 43, 45
Interval of approximation, 41
Invariance of the equations, 53
Inverse conformal map, 84, 101–103
Inverse Fourier transform, 10, 49–51, 90
Inverse matrices, 127
Inverse of the Fourier transform, 13
Inverse transform, 6
Isobar lines, 61, 69, 72
Isobars, 59, 61, 69, 72, 73
Isobar surfaces, 69, 73
Isotropic properties, 54
Isotropy, 54, 69, 74
Iso-velocity, 66, 74
Iso-velocity surfaces, 66, 74
Iterative sequence, 33

K
Kernel, 5, 107, 108, 112, 113, 117, 121, 125, 129, 130, 140

L
Laplace matrix, 127
Laplace transform inversion, 43
Laplace transforms, 41, 42, 45, 47, 79, 107–109, 111, 113, 121, 122, 127
Laplacian, vi, 1
L^2-approximation, 12
Least square estimation, 71
Lemma, 3–5, 12–14, 16, 17, 20–23, 25–30, 34, 47, 56
Lie, 40, 41
Lie's transformation theory, 53
Linear system, 44

M
Main function, 55, 63, 79, 129
Map, vi, 41, 55, 56, 67, 79, 84, 85, 99–104, 106, 111–113, 122, 123, 131–139
Mapping, 33–35, 41, 56, 99

Mathematica, v, vii, 54–56, 59, 60, 64, 65, 75, 79–86, 88, 92, 96, 97, 99, 100, 103, 104, 108, 110, 124, 142, 147
MathLie, 53
Matrices, 39–43, 48, 55, 97, 98, 107, 111, 131, 136, 137, 142, 147, 125127
Matrix, 26, 42, 43, 45, 48–51, 55, 56, 89, 92, 97, 98, 107, 114, 123, 124, 127, 131, 132
  form, 26
  product, 89
Matrix-vector form, 42
Maximum, 67–68
Maximum value, 34, 35
Mesh points, 84
Methods for solving, 6, 37–51
Methods of approximating solutions, 37
Methods of approximation, v, 37
Monotonically, 15
Moving waves, 54

N
Natural Dirichlet boundary, 73, 77
Navier–Stokes, 1
  equation, 37, 53, 63
  function, 129–144
  initial-boundary value problem, 54, 129
  problems, 45, 65, 120
NavierStokes3D[], 55, 63–65, 79, 129, 130, 141, 144, 147
Network, 54, 129
Neumann-type iterative method, 6
Non-algebraic decay, 100, 101, 106
Non-equidistant distribution, 99
Norm, 10, 12, 19–21, 26, 51, 58, 63, 66, 68, 71, 75, 139, 142, 147
2-Norm, 19
Norm-bound, 21
Normed, 10
Notebook, 54
Notebook network, 129
Novel methods solving PDE, 45
Numerical methods, 37–51
Numerical results, 53, 54
Numerical solution, vii, 6, 53, 56
Numeric approximations, 54
Numeric approximations of infinity, 61
Numeric Laplace transform, 45, 47, 107
Numeric values, 55, 86, 147

O
One-to-one transformation, 38, 41
Operations, 21–31, 38–45, 47, 89, 107
Operator
  equation, 19, 26, 33
  form, vi, 38
Optimize, 119
Option, 63, 65, 99–101, 106, 129, 130, 144
Orthogonal polynomial, 42
Oscillations, 66, 68
Outer product, 85, 87

P
Parallel
  computations, 55
  frame, 90
  implementation, 89, 92
Parseval's theorem, 10
Partial differential equations (PDE), v, v, vi, 1–7, 44, 45, 55, 56
Pascal's law, 54
PDE problem, v, 1, 5
Periodic extension, 13
Periodic functions, 38
Picard-like iteration, v
Piecewise analytic, v
Piecewise defined function, 99, 104
Point symmetries, 53
Poisson problem, 109, 110
Poisson's equation, 60, 79, 82, 83, 108–110, 117
Polynomial rate, v
Polynomials, 37, 41, 42, 59, 84
Positivity, 17, 42
Potential function, 59, 61, 82, 117, 119
Potentials, 23, 55, 59–61, 82–84, 97, 117, 118, 120
Practical applications, 54, 56, 59, 99
Pressure
  approximation, 61, 77
  calculations, 82
  integral, 75, 77
  strength, 58
Processors, 54, 55
Proofs, v, 3–5, 10–13, 15–18, 20–25, 27–30, 33–35, 40, 42, 63

Q
Quadratic, 34
  polynomial, 59
  transformation identity, 24
Quadrature, 42, 75

R
Rapidly convergent solutions, vi
Rate of convergence, 55

Rational functions, 37
Real line, vi, 1, 13, 99
Real parts, 42, 143
Relative measure, 70
Replacement rules, 64, 66, 142, 147
Rotational strength, 58
Rotations, 53, 57, 58

S
Scale-dilatation, 53
Scale-dilatation invariance, 54
Scaled Sinc function, 96, 97
Scaling, 54, 77, 129
Scaling structure, 54
Semi-infinite, 37, 99–101, 105
Sequence, 33, 34, 58, 66, 79, 102
Sequence of vector functions, 33
Shifted Sinc functions, 79, 103, 104
Shifting index, 103
Similarity solutions, 54
Sinc, v, 11, 37, 54, 79, 147
  approximation, v, 12–18, 38, 40–42, 45, 46, 61, 62, 67, 70, 74, 96, 102–104, 106, 107, 120, 147
  basis, 103–105
  collocation, 60, 120, 130
  convolution, 49, 50, 53–77, 107, 119, 129
  convolution integrals, 107
  functions, 12, 37, 79, 96–98, 103
  grids, 55
  indefinite convolution, 47
  interpolation, 38
  points, 41, 48, 55, 56, 60–63, 65–72, 75–77, 79, 84–92, 102, 104, 111, 112, 119–122, 124, 129, 131, 142–144
  quadrature, 42, 75, 77, 107
  series, 41
  spaces, 11, 41
  spaces are dense, 11
Sinc-Fourier transforms, 55
Sine integral, 97
Single machine, 54, 129
Singular integral equation, 44
Singularities, 43, 45
Solution of the N–S problem, 19–31, 141
Space of functions, v, 19
Spaces, v, 9–31, 33, 34, 39–41, 45, 46, 48, 57, 58, 60, 61, 66, 71, 73, 81–83, 92, 108–111, 119, 121, 133, 135–138
  of analytic functions, v, 9–18
  S^n, 11
  of solution, 19–31
Spatial and temporal discretization, 63
Spatial discretization, 85–88
Spatial distribution, 58, 62, 69, 71, 72, 74, 76
Spatial domain, 56
Spatial temporal convolution, 92–96
Special solutions, 53
Spectral data, 121, 124, 125, 135, 136
Spectral properties, 111, 113, 120, 121, 131, 132
Spherical coordinates, 6
Splines, 38
Splitted convolution integrals, 120–129
Splitting property, 124
Standard conformalmaps, 99
Standard methods, 55
Step length, 53, 60, 63, 65, 73, 84–92, 102, 103, 105, 111, 119, 129–131
Step lengths distribution, 88
Stream lines, 58, 59
Sub-domains, 108
Successive approximations, 7
Sup, 10, 12, 13, 16, 18, 20, 27, 71
Surfaces of constant velocities, 58, 67, 68, 74, 75
Symbolic and numeric calculations, 55
Symbolic calculation, 55, 107
Symbolic discrete representation, 85
Symbolic Laplace transform, 79, 107, 111
Symbolic/numeric functions, 79
Symbolic representation, 64, 80, 85, 86, 107, 127
Symmetries, 17, 53, 54, 58, 98
Symmetry transformations, 53
System of differential equations, 3

T
Telescoping sum, 16
Temporal-spatial convolution algorithm, 55
Theorem, 3–5, 10–12, 16–18, 20–23, 25, 27–30, 33–35, 39, 40, 42, 44, 47, 55, 56, 59, 70, 79, 82, 108
Töplitz matrices, 97, 98, 107
Töplitz matrix, 97, 98
Tori, 67, 68, 74, 75
Total vector field, 58, 59, 66, 68, 75
Translation invariant, 42
Trigonometric functions, 37
Trigonometric wavelet approximation, 40
Trigonometric wavelet formulas, 38
Truncated Sinc expansion, 38
Typewriter key, 41

U
Unbounded, 20

Uniformly bounded, 22
Uniformly bounded function, 10
Unique solution, 26, 33, 34, 44

V
Vectors, 1, 4, 6, 9, 19–21, 28, 29, 33, 38, 42, 43, 46, 49, 50, 56–59, 62–64, 66, 68, 70, 72, 73, 75, 80–83, 85–87, 117, 118, 120, 129
  field, 56–59, 62–64, 66, 68, 70, 72, 73, 75, 80–83, 85–87, 117, 118, 120, 129, 142–144, 147
  field plot, 82
  functions, 19, 20, 33
  spaces of functions, 9
Velocity, v, vi, 1, 56–58, 62, 66–68, 72, 74, 75, 77, 80, 82, 83, 129, 135–144
  components, 62, 64, 66, 67, 69, 70, 74, 76
  strength, 68
  vector fields, 18, 56, 68, 82, 83, 85, 117, 120, 142, 147
Viscosity, vi, 1, 53, 63, 65, 129
Vorticity, 56–58
Vorticity strength, 58

W
Wiener–Hopf, 44
  integral, 44
  method, 44
  problems, 44